A TRIDENT SCHOLAR PROJECT REPORT

NO. 230

"Employing Digital Signal Processing for Acoustical Analysis"

UNITED STATES NAVAL ACADEMY
ANNAPOLIS, MARYLAND

REPORT DOCUMENTATION PAGE

Form Approved
OMB no. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	9 May 1995	

4. TITLE AND SUBTITLE: Employing digital signal processing for acoustical analysis

5. FUNDING NUMBERS

6. AUTHOR(S)

Brannen G. McElmurray

7. PERFORMING ORGANIZATIONS NAME(S) AND ADDRESS(ES)

U.S. Naval Academy, Annapolis, MD

8. PERFORMING ORGANIZATION REPORT NUMBER: USNA Trident report; no. 230 (1995)

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

Accepted by the U.S. Trident Scholar Committee

12a. DISTRIBUTION/AVAILABILITY STATEMENT

This document has been approved for public release; its distribution is UNLIMITED.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

In the field of communications, digital signal processing based systems offer distinct advantages over comparable analog systems. Because analog systems have a tendency to change over time, they are inflexible and require constant attention. Digital based systems are stable and allow a great deal of flexibility. These advantages, as well as the advances in processor speed, enable digital based systems to surpass the accuracy, precision, and reliability of their analog counterpart. In underwater detection applications, digital signal processing techniques advance the level of sophistication in sensors and analysis, aiding in the detection of narrow band noises created by man-made sources like a submarine's machinery and propeller. With today's technology, it is possible to design receiving systems that capitalize on the digital advantages. In this project, digital signal processing was employed for the acoustic analysis of a sonobuoy's transmitted signal. The aim of the project was to construct a receiving system based solely on digital signal processing (DSP) that performed real-time extraction operations on the components of a common, fleet sonobuoy's received signal. These components contained the necessary information to resolve a contact's direction and sound signature. The system was provided a signal from a sonobuoy's receiver that simultaneously contained five signal components. The system extracted sound from the signal's left, right, and omni-directional sources. The processed sonar information was displayed on an oscilloscope and spectrum analyzer as well as audible through a speaker system.

14. SUBJECT TERMS: digital signal processing; acoustic; sonobuoy; digital filtering

15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED

U.S.N.A --- Trident Scholar project report, no. 230 (1995)

"Employing Digital Signal Processing for Acoustical Analysis"

by

Midshipman Brannen G. McElmurray, Class of 1995
United States Naval Academy
Annapolis, MD 21412

Certification of Advisor Approval

Assistant Professor Carl E. Wick
Department of Weapons & Systems Engineering

5/9/95

Acceptance for the Trident Committee

Professor Joyce E. Shade
Chair, Trident Scholar Committee

9 May 95

i. ABSTRACT

In the field of communications, digital signal processing based systems offer distinct advantages over comparable analog systems. Because analog systems have a tendency to change over time, they are inflexible and require constant attention. Digital based systems are stable and allow a great deal of flexibility. These advantages, as well as the advances in processor speed, enable digital based systems to surpass the accuracy, precision and reliability of their analog counterpart.

In underwater detection applications, digital signal processing techniques advance the level of sophistication in sensors and analysis, aiding in the detection of narrow band noises created by man-made sources like a submarine's machinery and propeller. With today's technology, it is possible to design receiving systems that capitalize on the digital advantages.

In this Trident project, digital signal processing was employed for the acoustic analysis of a sonobuoy's transmitted signal. The aim of the project was to construct a receiving system based solely on Digital Signal Processing(DSP) that performed real-time extraction operations on the components of a common, fleet sonobuoy's received signal. These components contained the necessary information to resolve a contact's direction and sound signature. The system was provided a signal from a sonobuoy's receiver that simultaneously contained five signal components. The system extracted sound from the signal's left, right and omni-directional sources. The processed sonar information was displayed on an oscilloscope and spectrum analyzer as well as audible through a speaker system.

Keywords: **Digital Signal Processing, Acoustic, Sonobuoy, Digital Filtering**

ii. ACKNOWLEDGMENTS

I would first like to express my sincere thanks to my advisor Professor Carl E. Wick. With patience and a guiding hand, he supported me throughout the process - from inception to conclusion. I will always appreciate his knowledge, experience and character. I would also like to thank my family - David & Carolyn Stroup who through midnight pep talks pushed me through the finish line.

iii. TABLE OF CONTENTS

Section 1: Background

The United States Navy uses the collection and evaluation of underwater acoustic energy as its primary method for detecting enemy submarines. One of the most prolific collection systems involves the use of sonobuoys. These air-dropped, acoustic detection devices may operate independently or in a complementary manner. Some may be deployed to attain initial detection of a possible submarine target while others provide shorter-range, more precise data on a target's range and bearing. These systems seek out submarine flow noise, i.e., the water flow over a submarine's hull, which is a gross, broad-band noise. They also seek out narrow-band noise, i.e., the noise produced by a submarine's machinery and propellers, which is very short in duration against slow-moving submarines and requires more sophisticated sensors and analysis [3]. Digital signal processing techniques provide a method for increasing sensor sophistication and analysis relative to comparable analog systems.

From the collected acoustic signals, a receiving & detection system extracts a contact's sound signature as well as its direction. The first step, however, is to separate the received signal into its component parts. Analog solutions to this problem have a tendency to be sensitive to the environment, change over time, and require constant monitoring. These solutions also tend to be inflexible, making it extremely difficult to implement new techniques. Digital solutions, however, are relatively stable with time. Their flexibility and structure allow new techniques and design parameters to be incorporated rapidly and easily.

The SPARTON AN/SSQ-53B is an example of a common, fleet sonobuoy that relays all sound signals in the audio frequency range heard in its vicinity. This information is

received by a cable-suspended hydrophone system and then transmitted via a VHF, FM radio to a receiver carried aboard anti-submarine warfare platforms such as helicopters or surface ships.

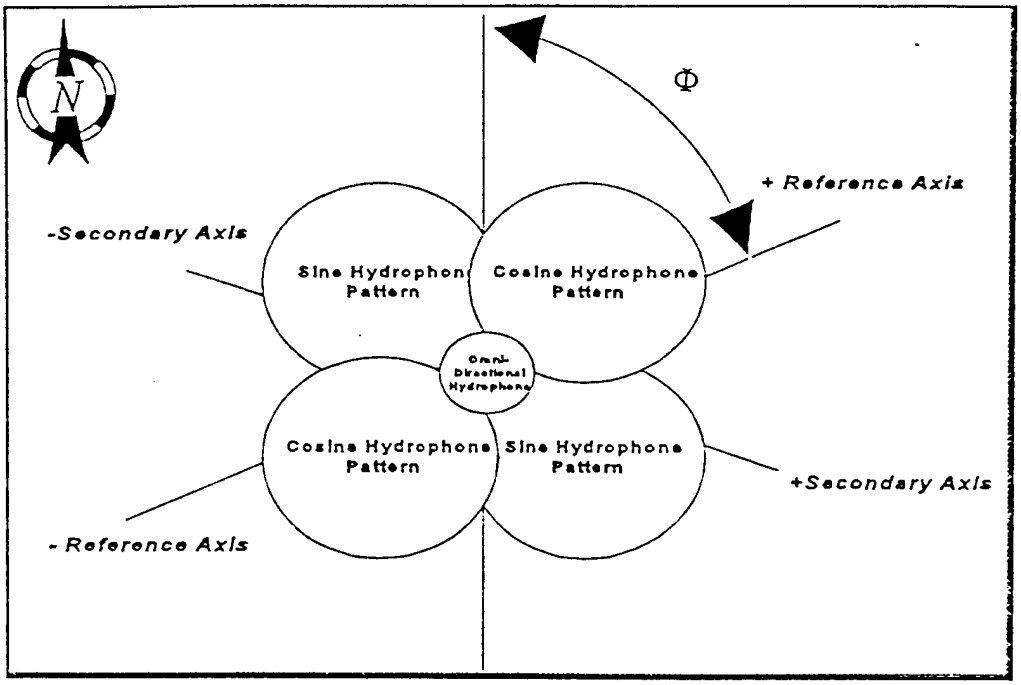

Figure 1 - The sonobuoy's acoustic senor array. The picture displays the two orthogonal directional hydrophone patterns as well as the omni-directional hydrophone. The clockwise angle from magnetic north to the reference axis is represented by an angle ϕ. The phase of the cosine channel subcarrier leads the reference phase by 90 degrees plus ϕ. The phase of the sine channel subcarrier leads the subcarrier of the cosine channel by 90 degrees.

The directional, acoustic sensor array of a common, Navy sonobuoy is composed of two sets of two hydrophones oriented along orthogonal axes. (See Figure 1.) One axis, the cosine axis, serves as a reference. Since there is no way to predetermine the orientation of the sonobuoy, a magnetic compass is also installed in the device. The compass is used to determine the orientation of the reference axis. The other orthogonal axis, the sine axis, is

oriented ninety degrees from the reference axis. This arrangement creates four quadrants in which a sound source can be located. The signal's direction is determined from the relative amplitude of the sine and cosine components of the information band centered at 15 kHz.

Acoustic signals cannot be radiated directly from the ocean to a receiver. They must be impressed on a very high frequency electromagnetic *carrier* waveform. **(See Figure 2.)** This is known as modulation. The modulation process may also be used to send several signals simultaneously. For example, the signals from the orthogonal hydrophone array are modulated. The modulated signal is then transmitted by radio link to an anti-submarine warfare platform. To recover the original signal, a receiver system demodulates the received signal. Fourier methods of signal analysis provide the most lucid illustration of the modulation and demodulation process.

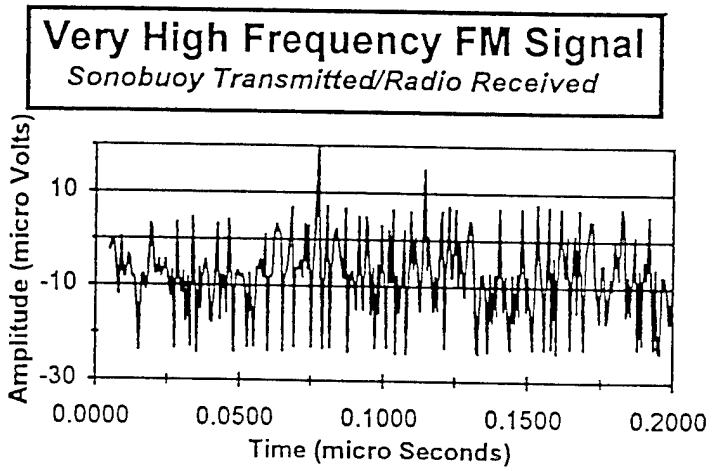

Figure 2 - A time domain representation of the sonobuoy's very high frequency, frequency modulated signal. The VHF, FM signal carries five superimposed signals to include the omni-directional signal, the frequency pilot, the phase pilot and the sine & cosine channels.

To shift the frequency of the sound signal received by the acoustic sensing array up in frequency, a system uses the modulation property of the Fourier Transform. Fourier theory states that multiplying a time function by $\exp(j\omega_o t)$ causes its spectral density to be translated in frequency by ω_o rad/sec. Using the Euler identity for cosine, a signal multiplied by $\cos(\omega_o t)$ in the time domain causes a frequency shift of $+\omega_o$ and $-\omega_o$ to occur in the frequency domain. The process of multiplying two signals together in the time domain is defined as modulation.

In general, modulation changes a parameter of one signal in proportion to changes in a second signal. In the previous example, the $\cos(\omega_o t)$ signal is modulated by the sound signal from the sonobuoy. Now that the frequency has been shifted up by ω_o, it can by efficiently sent by a transmitter and received by a receiver with a reasonably sized antenna, approximately 1 meter in this problem [8]. After being received by a receiver, this modulated signal must be demodulated in order to extract the original sound signal [9].

The recovery of the original signal requires another shift in frequency back to the signal's original position in the frequency domain. This is known as demodulation. Again, a system uses the modulation property of the Fourier Transform. The multiplexed signal from the receiver is multiplied by the Euler form of $\cos(\omega_o t)$ that shifts the signal to its original position and $+2\omega_o$. To discard the unwanted frequency components above the original signal frequency spectrum components, a system passes the signal through a lowpass filter [7].

To successfully demodulate a signal, the correct phase and frequency of the carrier, which is $\cos(\omega_o t)$ in the previous discussion, must be known. Incorrect frequency or phase errors cause erroneous results using the modulation property of the Fourier transform. Therefore, the frequency and phase must be the same in the transmitter and receiving system.

The task of ensuring that the frequency and phase are the same in the transmitting and receiving system is accomplished by using an oscillator that provides a synchronous signal between the transmitter and receiver. This technique is called synchronous, or coherent, detection [9].

Another method used to maintain synchronization between modulator and demodulator is by a pilot carrier system in which a sinusoidal tone, whose frequency and phase are related to the carrier frequency, is transmitted along with the information. This tone, which is outside the bandpass of the modulated signal, is detected and translated to the proper frequency to correctly demodulate the double sideband - suppressed carrier signal [9]. Duality between the time domain and the frequency domain gives the ability to use similar techniques in both domains.

The aim of this project was to construct a system based solely on Digital Signal Processing (DSP) that performed real-time extraction operations on the components from a common, fleet sonobuoy's received signal. This project focused on the **real-time** application of combinations of digital filters and other digital signal processing techniques to properly demodulate the incoming signal and remove the five information components. The receiver system performed a time domain analysis using a point-by-point, real-time filtering scheme that extracted the five essential components from the received signal.

Each of the signal's components was extracted using digital filters [3]. **(See Figure 3.)** The first component was the omni-directional signal, which was derived from information received by the omni-directional hydrophone. The receiver system extracted the omni-directional signal, which provided audible information as well as the information to resolve

directional ambiguity. Information from the directional hydrophones was carried as sidebands on a combined signal. This signal was separated into its component parts. From the component parts, a detection system can determine compass heading and contact bearing.

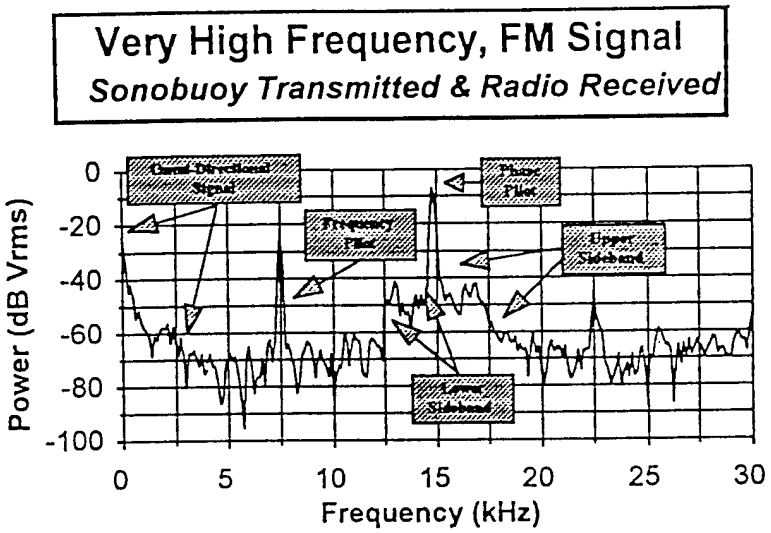

Figure 3 - A frequency domain representation of the VHF, FM signal. The frequency domain provides a much more lucid illustration of the spectral composition of the received signal.

The sonobuoy also transmits two pilot signals - a frequency pilot and a phase pilot. To obtain a contact's location, a detection system must also know the position of the sonobuoy's array of acoustic sensors in the water. A detection system determines the sonobuoy's position with a signal derived from the sonobuoy's internal magnetic compass. The magnetic compass changes the phase of the frequency pilot relative to the phase pilot. The receiver system extracted the pilot signals, which resolved this information. From these two pilot signals, the receiver system had the necessary information to demodulate the FM signal and extract the crucial acoustic information that allows a detection system to resolve ambiguity.

There are several ways to extract the signal's components. Analog methods could be used to build a series of circuits. With resistors, capacitors and inductors, a system may be constructed using analog lowpass filters, bandpass filters, and highpass filters. These filters, in combination with analog components such as double-balance mixers and a phase lock loop, can separate the combined signal into its component parts. The advantage of analog systems rests primarily in the considerable knowledge and experience amassed concerning their implementation and design.

A system using digital methods can also extract the signal's components. Unlike analog methods, digital methods are algorithms and software implemented on digital signal processing boards. The advantages that digital solutions have over analog solutions are the following: smaller size systems, much lower component tolerances, greater accuracy, greater reliability, and multitasking - the ability of certain components to share filtering tasks. The disadvantages of these digital based systems are the problems that we encounter due to the limitations of processor speed. If the processor is not fast enough, then there is insufficient time to accomplish the necessary operations before the next piece of sampled data arrives to be processed. This causes several significant problems. One problem is aliasing, which is signified by spectral overlap. Another problem is quantization errors, which are signified by a lack of resolution. With advances in electronics, processors' speeds and sampling rates are increasing. This combination in addition to anti-aliasing filters aid in eliminating the effects of aliasing. The number of levels that a signal can be sampled is increasing, which improves system resolution [11]. Because of these advances in technology, digital filtering components are becoming more prevalent in modern day systems.

Section 2: Experimental Apparatus & DSP

A sonobuoy consists of two distinct components - a transmitter system and the acoustic sensing array. In this research, a SPARTON AN/SSQ-53B, a common fleet sonobuoy, was modified to use the acoustic sensor array and onboard transmitter. The SPARTON AN/SSQ-53B was removed from its holding canister and the acoustic sensor array was deployed. The array consisted of four hydrophones as well as an omni-directional hydrophone encased in a plastic that matched the impedance of ocean water. A long cable connected the array and the transmitter. During normal deployment, this adjustable cable allows the array to collect information at a pre-set depth while the transmitter floats on the surface. **(See Figure 4.)**

The array was suspended in air from a platform to simulate its deployed position. The platform could rotate around its base to simulate the sonobuoy's change in position in the water. The acoustic sensing array collected the acoustic energy from the surrounding air environment, which it sent through the cable to be transmitted to the receiver system. The transmitter then superimposed the acoustic information and other reference information onto a very high frequency carrier signal. At the receiver system, each of the signal's components were separated and evaluated using different digital signal processing techniques.

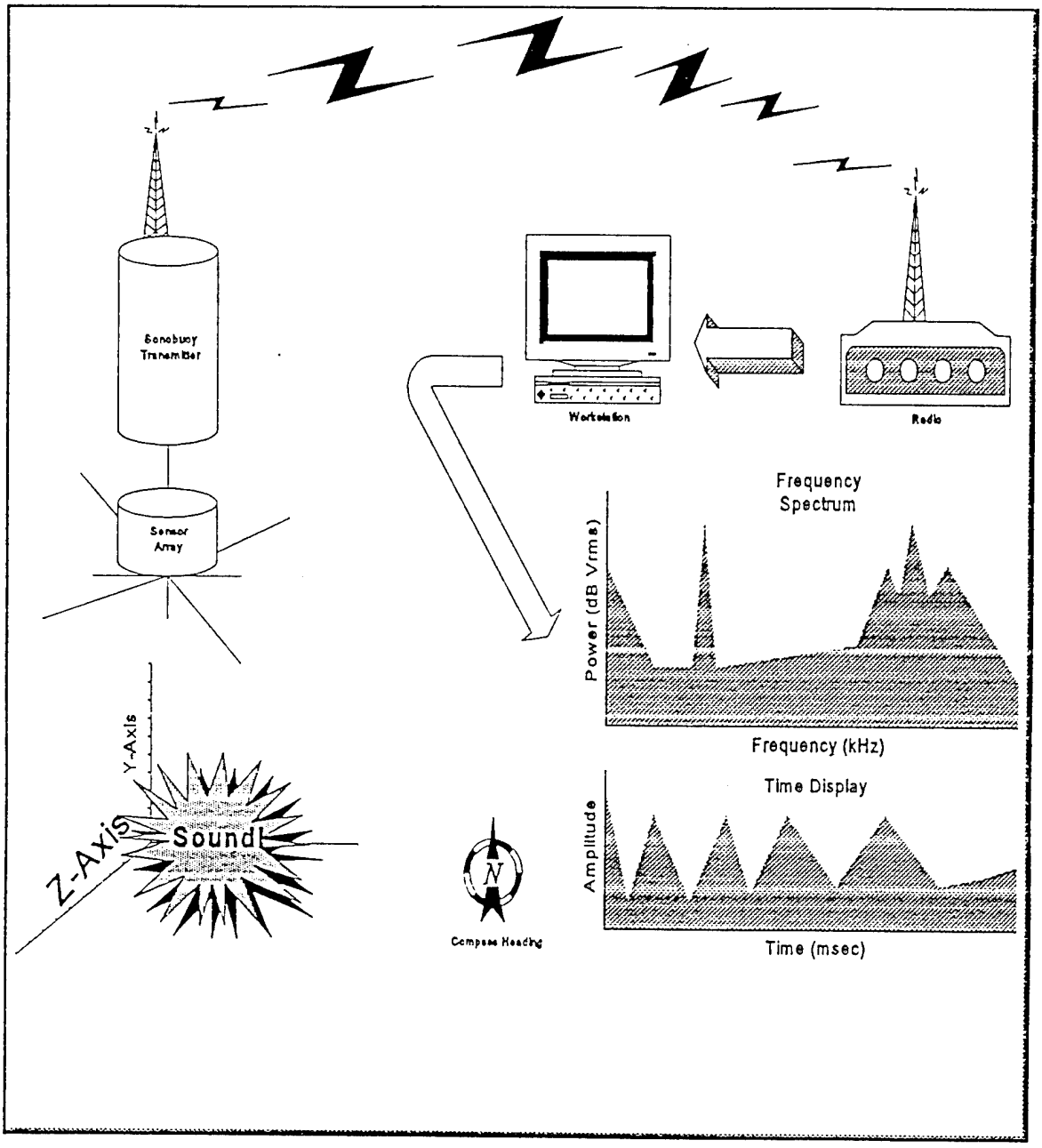

Figure 4 - A system flow diagram of the experimental apparatus. The sonobuoy collects the acoustic information through its acoustic sensor array. A radio receives the sonobuoy's transmitted signal. The signal is patched to a computer equipped with a digital signal processor. The processor contains algorithms that demodulate the signal. It is then displayed in the time and frequency domain for analysis.

The receiver system consisted of a radio receiver routinely used onboard an anti-submarine warfare helicopter, an IBM compatible personal computer equipped with the AT&T Digital Signal processing board, a spectrum analyzer and an oscilloscope. The radio received the very high frequency, FM signal transmitted from the sonobuoy. With a coaxial cable and plug connector, I took the received signal from the radio receiver and connected it into an onboard input channel of the Digital Signal Processor(DSP). The continuous signal was quantized by the DSP's onboard analog-to-digital converter. The received signal was then processed using algorithms downloaded onto the DSP from the IBM compatible personal computer.

An IBM compatible personal computer was used to interface and control the digital signal processor, which is mounted in an expansion slot of the PC. The personal computer was equipped with support software and the tools to communicate with the AT&T Digital Signal Processing Board's **independent processor**. Using the personal computer and software development packages such as Borland C++, a program was created and downloaded onto the DSP board. The program contained the algorithms and other necessary commands to properly demodulate the signal and extract the information components. Once downloaded and initiated on the DSP board, the program performed the necessary algorithms on the incoming signal to separate the signal's information components. The information gathered by the DSP was sent out of the digital signal processing board through two digital-to-analog converters. The converters transformed the discrete digital signal from the DSP into a continuous signal that was displayed and interpreted on equipment such as a spectrum

analyzer and oscilloscope.

The processed signal was connected to a spectrum analyzer, an oscilloscope and an audio amplifier. The spectrum analyzer displayed the signals in the frequency domain. The spectrum analyzer displayed two signals, which corresponded to the two output channels of the DSP. This was an extremely useful analysis because of the ease that one may visually inspect the occurrences of the signal's spectral or frequency components. These were the tones and information that the acoustic sensor device received. The oscilloscope allowed the user to see the phase shifts of the received signal's different components. The phase shift is an important ingredient in determining the sound contact's line of bearing. The amplified audio signal is significant to trained operators who can match the sound with a sound source. From the combination of information from these displays, a detection system is able to locate the bearing of a sound source as well as identify its sound signature.

Section 3: Digital Filtering

Digital filtering is one of the most frequently used techniques in digital signal processing. In this research, the receiver system relied heavily on digital filtering to extract information from the received signal, which contained multiple components. The technique evolved from the simulation of analog filters on digital computers [11]. In the mid-1960s, digital filter design began to come into its own as a discipline. Several factors contributed to the process's early success. Faster and less costly computers as well as improvements in integrated circuit design played a major role in the increased use of DSP-based components.

It became apparent that digital filters were becoming competitive with analog filters for real-time signal processing [11].

Though analog filters consist of resistors, capacitors and inductors, digital filters are software programs that implement difference equations. Difference equations describe mathematical operations that consist of a series of add & multiply operations. These operations serve to manipulate an incoming signal so that its spectral content can be altered. Digital filters are most often used to enhance signals by removing from them the unwanted spectral components. An example is the *frequency-selective filter or band-pass filter,* which is the one of the most frequently implemented type of digital filter. It is designed from frequency response specifications that allow only certain frequency bands to pass through.

Digital Filtering Systems may be *nonrecursive* or *recursive*. Nonrecursive systems are memory-less or instantaneous. **(See Equation 1.)** The instantaneous output depends only on the instantaneous input. **(See Figure 5.)** The second type of systems are recursive or dynamic systems. **(See Equation 2.)** In these systems, the input depends on the instantaneous inputs and the past outputs [6]. **(See Figure 6.)**

$$y(k) = \sum_{i=0}^{m} B_i x(k-i) \qquad (1)$$

Equation 1 - The **x** represents the discrete time input. The **y** represents the discrete time output. The **B** represents the set of coefficients applied to the input, which represents the delay factors [4].

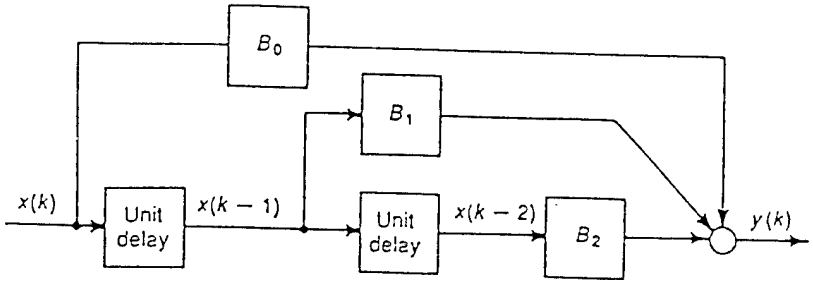

Figure 5 - The equation and figure represent the general form of the non-recursive or finite-memory system. The non-recursive system model provides no feedback. The instantaneous outputs depend only on the instantaneous inputs [11].

$$y(k)=\sum_{i=1}^{n} A_i y(k-i) + \sum_{i=0}^{m} B_i x(k-i) \qquad (2)$$

Equation 2 - The **x** represents the discrete time input signal. The y represents the discrete time output signal. The **A** represents the set of coefficients applied to the input for the delay factors. The B represents the set of coefficients applied to the output for the delay factors [4].

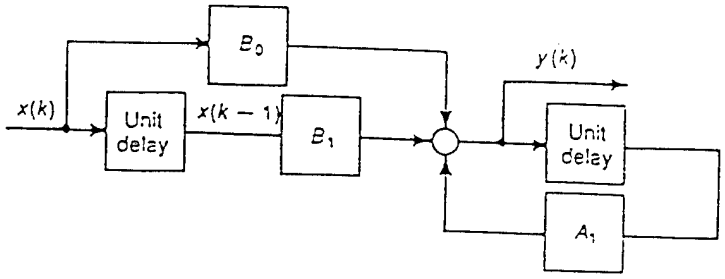

Figure 6 - The equation and figure represent the general form of the recursive or infinite memory dynamic systems. The recursive system model incorporates feedback. The instantaneous outputs depend on past outputs as well as instantaneous inputs [11].

This research focused on linear time - invariant (LTI) filters that can be implemented in a nonrecursive or recursive form. This LTI class of filter operates on a signal that is a function of some independent variable or variables. The signal may be a *continuous-time signal* or an analog signal x(t) that is a function of the continuous variable t, which represents time. The signal may also be *a discrete-time signal* or discrete signal x(kT) that is defined only at discrete time instances t=kT [10].

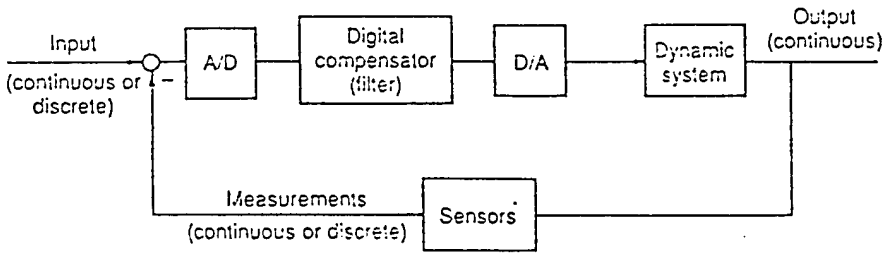

Figure 7 - Digital components can be incorporated into a system that operates on continuous signals. A continuous signal may be quantized by a analog to digital converter. The quantized signal is processed digitally and the output is transformed back to a continuous signal via a Digital to Analog Converter [11].

Signals are sampled with an analog-to-digital (A/D) converter. (See Figure 7.) A continuous signal, when sampled at regular intervals by this method, is transformed into a discrete signal consisting of a sequence of sample values of the original continuous signal. If the sampled signal is to be processed in a digital computer, its sampled values must be represented by a number of binary digits. The number of bits used is dependent on the capability of the A/D converter. Consequently, only a finite number of amplitude levels is

possible in a sampled system. **(See Figure 8.)** The difference in the sampled value and the actual value is termed quantization error.

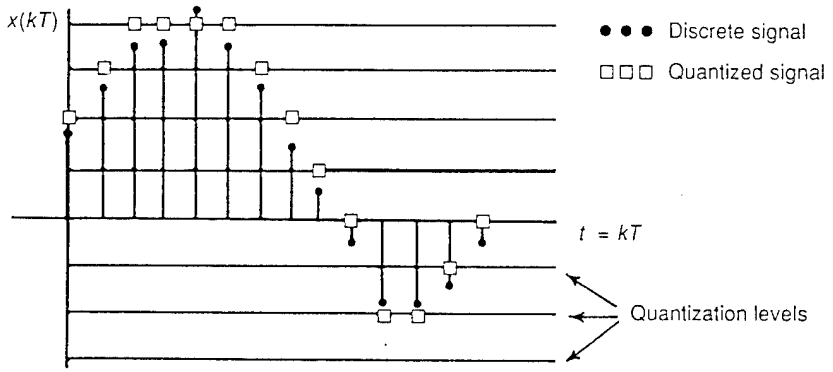

Figure 8 - In order for a computer to process a continuous signal, the signal must be quantized. The sampled signal is represented by a series of bits corresponding to the appropriate quantization level. The difference in the sampled value and the quantized value is called quantization error [11].

A digital signal represents this quantized discrete signal $x_q(kT)$. Processed digital signals can be converted back to continuous form by means of a digital-to-analog (D/A) converter. The device transforms the discrete signal to a continuous signal [5]. **(See Figure 9.)**

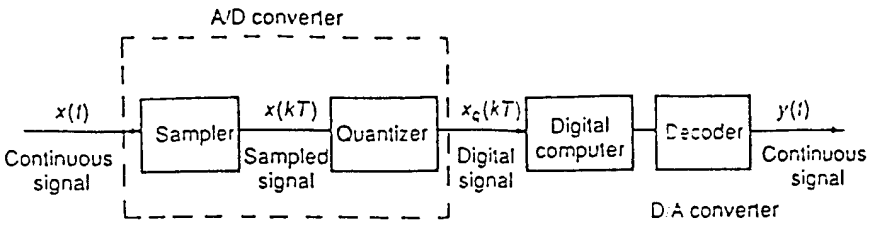

Figure 9 - A continuous signal is sampled then quantized. The quantized signal is processed by the digital computer. The decoder transforms the digital output to a continuous signal [7].

In the time domain much of digital filtering is based on the computational algorithm that is referred to as the linear constant coefficient difference equation. (See Figure 10.) Once a continuous signal has been quantized, the elements of the signal are placed into a difference equation whose coefficients represent designed frequency performance specifications. This equation represents the series of add & multiply operations that may be performed very efficiently by a Digital Signal Processor [1].

General Form of a Linear Difference Equation

$$a(0)*y(n)=\sum_{i=0}^{i=n-i}(x[n-i]*b(i)) - \sum_{i=1}^{i=m-i}(y[n-i]*a(i))$$

x: The discrete time input signal to the system represented by the filter
y: The discrete time output signal to the system represented by the filter
a: Set of coefficients applied to the input representing multiplying factors for delays
b: Set of coefficients applied to the output representing multiplying factors for delays

Figure 10 - The linear constant coefficient difference equation represents the series of add & multiply operations which can be performed very efficiently by a DSP. By applying designed coefficients, we use this equation to represent digital frequency selective or bandpass filters [1].

Using the linear difference equation as a basis, I designed two types of digital LTI filters. The first type of filter is termed a finite impulse response filter or (FIR). The FIR filter is characterized by a set of filter coefficients that alter the signal's spectrum when convolved. The filter has finite memory, because it forgets all inputs before the mth previous one [5]. Because there is no feedback, the filter cannot be unstable. This is sometimes a desirable property. For example, if a digital algorithm were used as a filter in a fire-control system, it would be an advantage for the filter to have finite memory for acquiring transients [2]. For this research, FIR designs were not used, because they require too much time for real-time applications.

The second type of filter is termed an infinite impulse response filter or IIR. The IIR filter is similar to a FIR except that the previous output samples are used to form the current output. This feedback creates the infinite impulse response. They have fewer terms, and consequently require less processing time. Because the IIR filter incorporates feedback, I took care in designing a filter that would not go unstable. The IIR filter had two advantages. Because processing speed is one of the main limiting factors in digital filter design, efficiency was a consideration. The IIR filter also offered a more accurate amplitude response than the FIR filters. In this application, amplitude information is an important ingredient in determining a contact's direction. Also, because IIR filters incorporated past outputs through feedback, transients could be used to enhance detection schemes.

In modeling digital filters, there are several techniques available for developing the proper linear difference equation to be used in the digital system. A continuous system can

be represented in the same manner as a discrete time systems. In the time domain, the filter can be modeled with a series of delays, where powers of z represent the increasing delay of a signal. **(See Equation 4.)**

$$Y(z) = \frac{b(1) + b(2)z^{-1} + ... + b(nb+1)z^{-nb}\ X(z)}{1 + a(2)z^{-1} + ... + a(na+1)z^{-na}} \qquad (4)$$

Equation 4 - The **x** represents the discrete time input signal. The **y** represents the discrete time output signal. The **a** represents the set of coefficients applied to the input. The **b** represents the set of coefficients applied to the output. The **z** represents the delays applied to the terms [3].

An IIR filter, termed an all-pole, recursive or autoregressive filter, is characterized by nb=0. When na=0, the filter may be called a Finite Impulse Response, all zero, nonrecursive, or moving average filter. If both na and nb are greater than zero, the filter may be called IIR, pole-zero, or recursive or autoregressive moving average [10].

There are several techniques widely used to design digital filters. The four most common are *direct FIR filter design, direct IIR filter design, Inverse Filter Design, IIR filter design using analog prototypes.* Direct FIR filter design and direct IIR filter design produce frequency selective filters that are designed from magnitude response performance specifications. Inverse filter design attempts to find the proper filter coefficients that are responsible for a given set of time-response or complex frequency-response data. IIR filter design using analog prototypes produces frequency selective filters that are designed from magnitude response performance specifications. This technique uses the common analog

models of Butterworth, Chebyshev, Inverse Chebyshev, and Elliptic to construct real time digital filters. This project used the direct IIR filter designs to obtain the various filter components that satisfied the filter performance specifications of the receiver system [6].

The various methods of filter design primarily differ in how performance is specified. This project implemented digital filters that were modeled using the common analog prototypes. The Butterworth, Chebyshev I, Inverse Chebyshev, and Elliptic filters can be used when the specific amounts of passband ripple, stopband attenuation, or transition width are defined. **(See Figure 11.)** These prototypes are very similar in their implementation in the digital and analog design space.

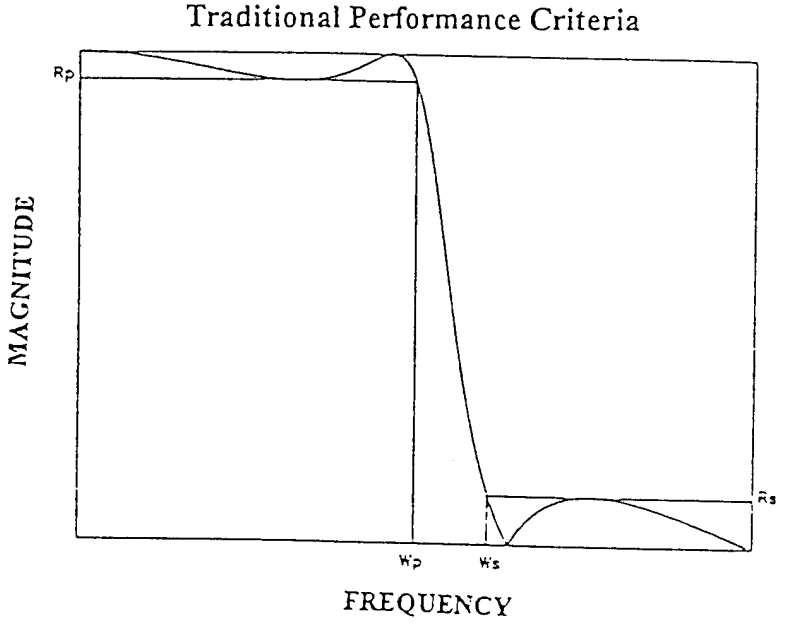

Figure 11 - The traditional performance criteria in filter design. These criteria are passband ripple, stopband attenuation, and transition width [11].

The specific type of filter used in a design depends on the type of performance required from the components and the hardware constraints. For example, the elliptic filter

will satisfy performance constraints (such as meeting a specific attenuation level at a given frequency) with a lower order filter than the other filter types. The price paid for this performance is the time to compute the actual filter. Differences such as these are vital in choosing the appropriate design for the realization of the system.

The Butterworth filter is a filter that amplifies the desired signal and attenuates any unwanted signal. The response of a Butterworth filter has a maximum deviation in passband and transmission at the passband only. The rolloff between the passband and the stopband is slow, so that a lower order Butterworth filter does not provide a good approximation of an ideal filter. The Butterworth filters sacrifice rolloff steepness for a monotonical increase in the pass and stop bands. This property makes the Butterworth response very flat near $\omega_o=0$ and gives the maximally flat response. As the order N is increased, the degree of passband flatness increases and the filter response approaches an ideal "brick-wall" type response [10].

Equation 5 represents an all-pole approximation to this ideal filter and its magnitude function.

$$M(\omega)= \frac{1}{(1+\omega^{2n})^{1/2}} \qquad (5)$$

Equation 5 - The ω represents the input frequency. The **M** represents the magnitude of the output [4].

$$H(z)= \frac{B(z)}{A(z)} = \frac{b(1)+b(2)z^{-1}+\ldots+b(n+1)^{-n}}{1+a(2)z^{-1}+\ldots+a(n+1)z^{-n}} \qquad (6)$$

Equation 6 - The **H** represents the transfer function of the filter. The **Bs** are the system outputs. The **As** are the system inputs. The **z** represents the delays on the coefficients [11].

The specific filter coefficients are computed from a bilinear transformation. **(See Equation 6.)** The poles of an Nth-order Butterworth filter can be determined from a graphical construction. The poles lie in a circle that has a constant radius and are spaced by equal angles of π/N, with the first pole at an angle $\pi/2N$ from the $+j\omega$-axis. Since the poles all have equal radial distance from the origin, they all have the same frequency [10].

The frequency response of the Chebyshev filter shows equal ripples in the passband and a monotonically decreasing magnitude response in the stopband. This type of filter has a much sharper rolloff than the Butterworth filters. The poles are evenly spaced about an ellipse in the left half-plane as defined by the following transfer function. **(See Equation 7.)**

$$\Omega(s) = \frac{C}{\prod_{k=1}^{n}(s-s_k)} \tag{7}$$

Equation 7 - The **C** represents the function gain. The bottom term represents the term summation of the frequency response [4].

The frequency response of the Inverse Chebyshev filter is similar to that of the Chebyshev filter except that the ripples occur in the stopband and the frequency response is flat in the passband. The Inverse Chebyshev filters are monotonic in the passband and equiripple in the stopband. The pole locations are the inverse of the pole locations of the Chebyshev filter, whose poles are evenly spaced about an ellipse in the left half-plane [10] **(See Equation 8.)**

The Elliptic filter should be used only if ripples are allowable in both the passband and the stopband. They offer steeper rolloff characteristics than the Butterworth filters and the

Chebyshev filters, but at the expense of pass and stop band ripple. These filters have the sharpest rolloffs for the same order when compared with the Butterworth or Chebyshev. Smaller values of the passband ripple Rp and larger values of stopband attenuation Rs both lead to wider transition widths (shallower rolloff characteristics)[10].

$$H(s) = \frac{W_o}{D_o(s)} \prod_{i=1}^{r} \frac{s^2 + C_i}{s^2 + A_i s + B_i} \qquad (8)$$

Equation 8 - The **H** represents the output over input relationship of the frequency response. The transfer functions values are determined by specific specifications [4].

Section 4: Digital Filtering Specific to DSP32

In this research, software was developed that properly demodulated the received signal from the sonobuoy and extracted the information frequency bands from the multiplexed signal. To accomplish this, the receiver system used a combination of lowpass, highpass, and bandpass digital filters as well as other signal processing techniques. To implement filters, program modules were designed to execute difference equations on the AT&T DSP32C, the digital signal processor. Due to the AT&T DSP32C's hardware specifications, computational time and resolution were two of the considerations in the software design.

The DSP can perform A/D conversions up to 51.2 kHz. Nyquist sampling theory dictated that in order to overcome the problems of aliasing, the incoming signal was sampled at a rate greater than twice the signal's highest frequency. Of the signals used in this research, the highest frequency the system used was approximately 25 kHz. The DSP board discarded any frequencies above 25 kHz with an anti-aliasing filter. In this application, the DSP32C ran

at 51.2 kHz. At this sampling frequency there were 19.531 μsecs between samples. Because a real-time, point-by-point analysis was used, this was the amount of time available in which to accomplish the manipulations on each point of the received signal.

The processor onboard the AT&T DSP32C operated at 12.5 million instructions per second. Using the 51.2 kHz sampling rate, the processor executed 244 instructions before the next sample arrived. For this reason, efficient code was developed for the demodulation and extraction process. Balancing the number of necessary operations with the amount of time for execution was a design necessity. In order to accomplish processes that require more than 244 instruction, the sampling rate must be reduced. This, however, limits the frequency bound of signal a receiver system can analyze.

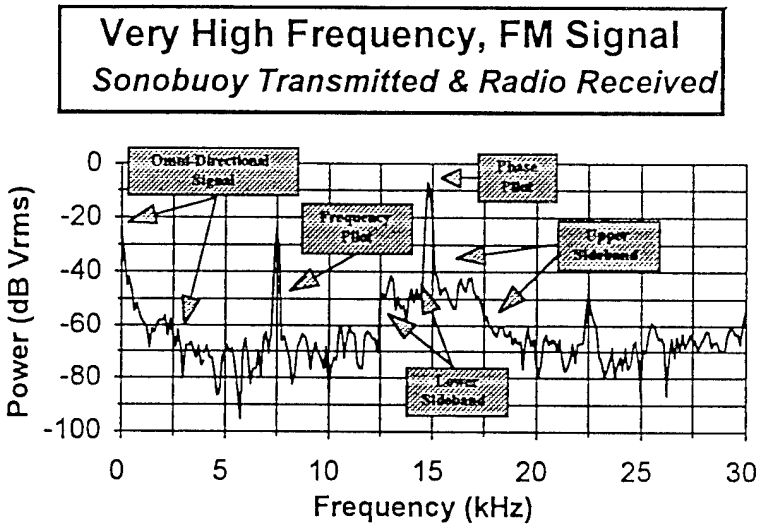

Figure 12 - The frequency spectrum representation of the sonobuoy's VHF, FM signal. The signal consists of five superimposed signals. Each of these signals must be extracted and processed to properly demodulate the received signal.

The final program considered the design limitations and implemented efficient modules which solved the problem on a piecemeal basis. The process began by defining the signals that carried information. **Figure 12** illustrates the five information components of the received signal. These included an omni-directional channel, a frequency pilot, a phase pilot, a sine channel and a cosine channel.

The first signal of interest was the omni-directional channel, which was the output of the omni-directional receiver. This signal was centered in the zero Hertz range and the input frequency may range from 10Hz-2600Hz. Using these ranges as filter performance specifications, a sixth order, lowpass, Butterworth digital filter with a cutoff frequency of 2600 Hz was constructed. This filter removed the spectral content above 2600Hz at a rate of 120dB per decade. The only signal that remained in the resulting spectrum after processing was the omni-channel signal. **(See Figure 13.)** The coefficients used in this and other filters may be found in **Appendix Section 9.2**.

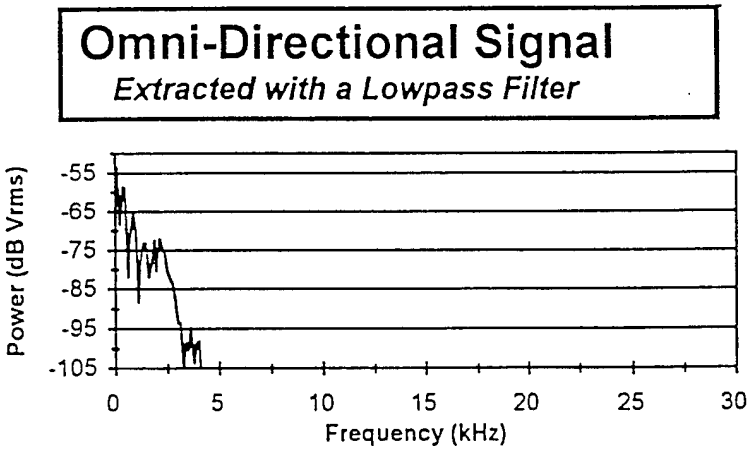

Figure 13 - The omni-directional signal is derived from the omni-directional hydrophone. It is extracted with a sixth order, Butterworth, lowpass digital filter.

Once the omni-channel was removed, the receiver system extracted the pilot signals. The frequency pilot had a constant amplitude and was centered at 7.5 kHz with a passband of 25 Hz. It was exactly one-half the frequency of the phase pilot. The frequency pilot was often useful as a reference signal because the phase pilot is obscured by the cosine and sine information. To remove the frequency pilot, second order, Butterworth, bandpass filter centered at 7.5 kHz with a passband of 50 Hz was used. The resulting spectrum contained only the frequency pilot. All the other signals were removed. **(See Figure 14.)**

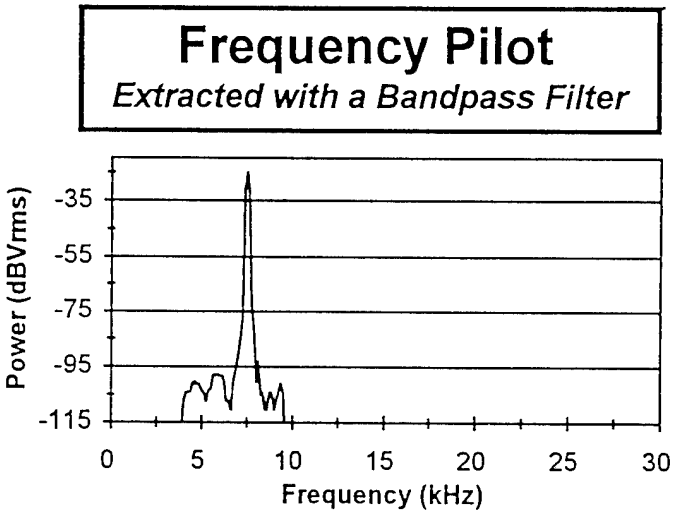

Figure 14 - The frequency pilot is extracted with a second order, Butterworth, bandpass digital filter.

The next signal removed was the phase pilot. The phase pilot was a constant amplitude 15 kHz phase reference signal with a passband of 100 Hz. The sine and cosine hydrophone channels composed its upper and lower sidebands. To extract the phase pilot, a second order, bandpass filter centered at 15 kHz with a passband of 20 Hz was used. The passband was limited to minimize the amount of unwanted spectral components from the

cosine and sine channels that composed the sidebands. The filter extracted the phase pilot and minimized the spectral contributions of the sidebands. **(See Figure 15.)**

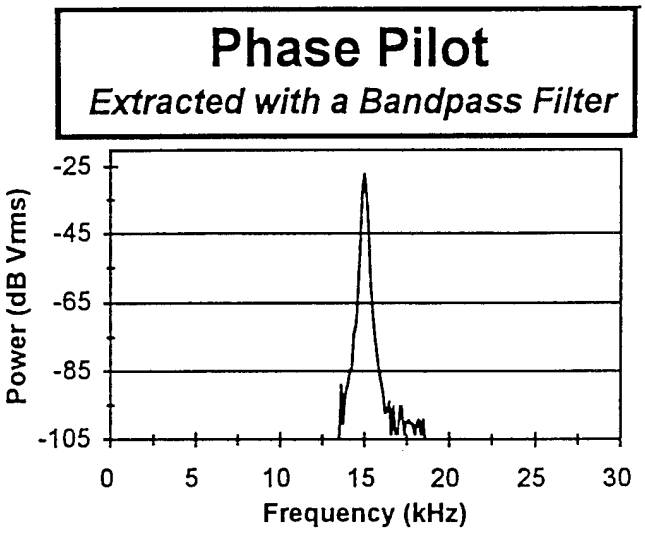

Figure 15 - The phase pilot is extracted with a bandpass digital filter.

The sine and cosine channels were the only remaining information signals. These signals were sidebands of the phase pilot. These signals were separated from the received signal to recover the directional information. These signals were retrieved with a three step process. First, the complete information band that contained the sine channel, cosine channel and phase pilot was isolated. To extract this band, a second order, bandpass, digital filter with a passband of 5.2 kHz was built. **(See Figure 16.)**

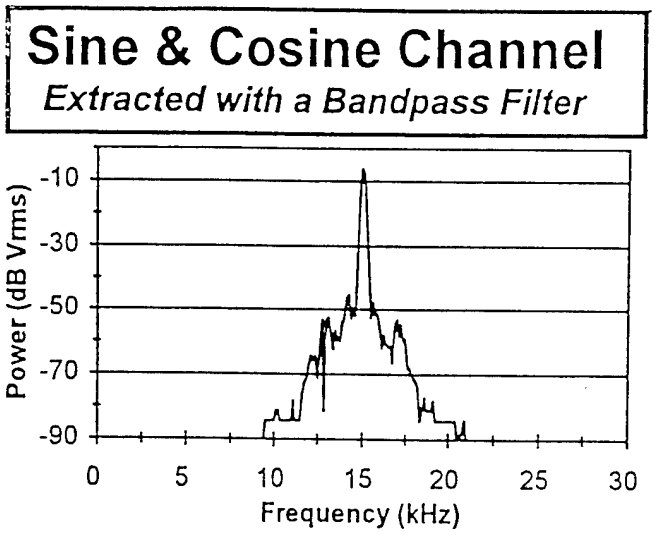

Figure 16 - The sine & cosine channels are extracted with a bandpass digital filter.

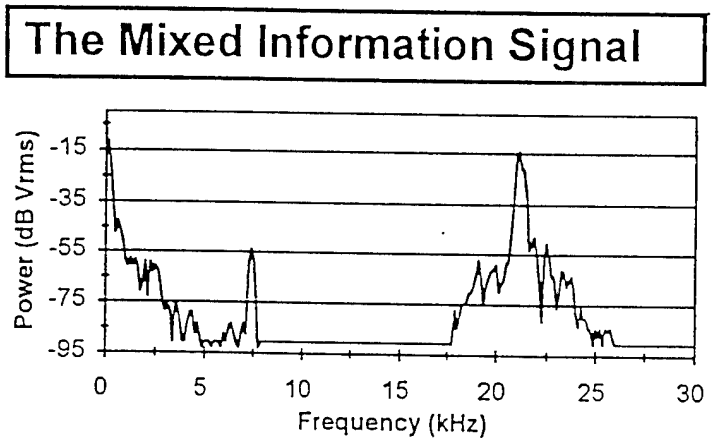

Figure 17 - The phase pilot and the information band consisting of the sine & cosine channels are mixed. This creates a copy of the signal at baseband and a copy of the original signal above the modulating frequency.

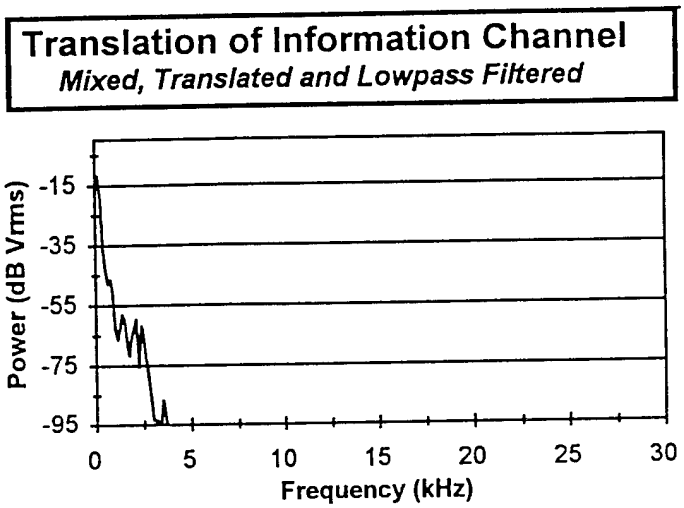

Figure 18 - The mixed signal is lowpass filtered. This leaves only the information sideband.

Next, the phase pilot alone, extracted in the previous step, was taken and mixed with the extracted upper and lower sideband components. The mixing process was a multiplication operation in the time domain that resulted in multiple frequency components. Mixing translated the information band to baseband (frequencies near zero). It also created a copy of the information band above 15 kHz. **(See Figure 17.)**

A fourth order, lowpass, Butterworth digital filter was used to remove the unwanted spectral components. This filter removed the spectral components above 2600Hz, leaving only the information sideband as baseband frequencies. **(See Figure 18.)** This receiver system performed real-time, point by point signal processing. Signal manipulations were performed in the time domain and the signal analysis was performed in both the frequency and time domain.

The baseband information represented one of the orthogonal channels. The other may be extracted by retarding the phase of the pilot and repeating the mixing process. The compass heading can be determined by examining the phase difference between the pilot signal and the phase reference. This was not accomplished in this project. Ambiguity between signals found in the orthogonal channels can be found by examining the phase of the baseband signals relative to the omni-directional channel. This was also not done in this project. These activities represent operations performed by a detection system and are possible future activities for continued study of this problem.

Section 5: System Performance

In this research, the techniques and theory used in the Digital Signal Processing based receiver system were successful and effective in solving the problem. The receiver system was able to properly demodulate the received signal in real-time. Using digital filters, the receiver system was successful in individually extracting the omni-directional channel, the frequency pilot, the phase pilot, the sine channel and the cosine channel from the received signal.

As illustrated in the graph shown in **Figure 19**, the processed omni-directional signal was successfully removed from the received signal using a sixth order, Butterworth digital filter. Although more effective, higher order filters were designed, a sixth order filter extracted the omni-directional signal with the minimum amount of computational time. The processed omni-channel retained its original shape and the filter attenuated the spectral components above 2600 Hz. **(See Figure 19.)**

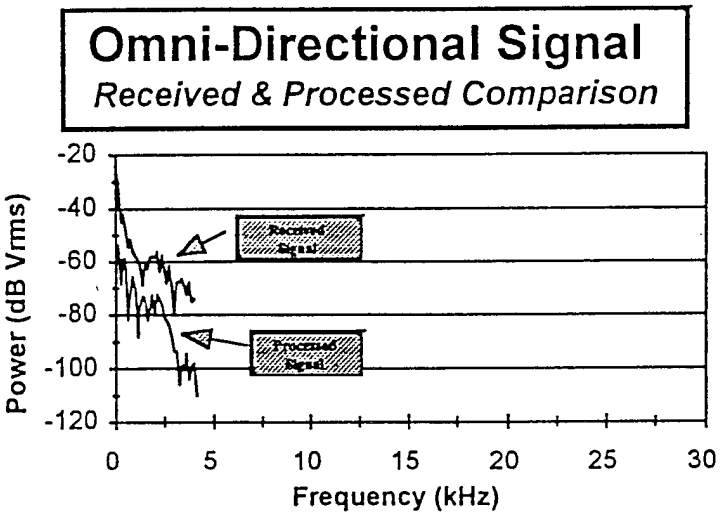

Figure 19 - Comparison of the received and processed signals. The omni-directional signal is extracted with a fourth order, Butterworth, digital filter.

The frequency pilot required a narrow, bandpass digital filter. Again due to limitations in computational time, the filter order was designed to allow time for the other processing operations. The second order, bandpass digital filter with a passband of 50 Hz was effective in removing the frequency pilot from the received signal in the minimum amount of time. The filter succeeded in attenuating the unwanted spectral components outside of the 50 Hz passband of the filter - leaving only the frequency pilot. (See Figure 20.)

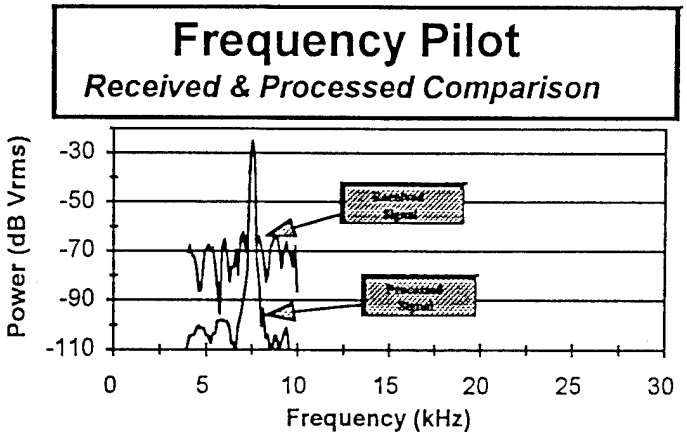

Figure 20 - The frequency pilot is extracted with a second order, bandpass, digital filter.

The information band centered at 15 kHz was the most difficult to extract successfully. In the receiving scheme, highpass filters were first used. A sixth order, Butterworth highpass digital filter was implemented. The performance of the highpass was adequate. **(See Figure 21.)** It adequately attenuated the unwanted spectral components to an acceptable level and computational time to compute the sixth order filter alone was within the hardware constraints. But, combined with the other necessary digital signal processing information, its computational time was unacceptable. For this reason, a second order, Butterworth, digital, bandpass filter was used to retrieve the information band. **(See Figure 22.)**

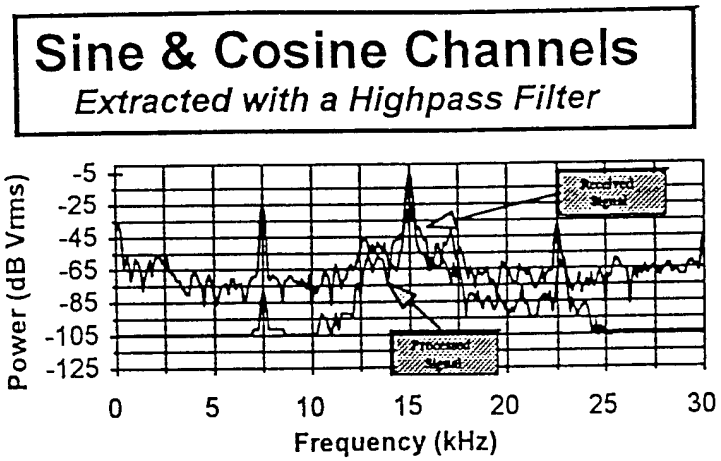

Figure 21 - Using a highpass filter was not the most effective way to attenuate unwanted spectral components and extract the desired signal.

Experimenting with the bandpass designs, acceptable filters were developed that could be combined and still be within the hardware constraints. With the phase-adjusted pilot, the receiver system successfully executed a mixing process coupled with a lowpass filter operation that translated the sidebands to baseband. (See Figure 23.)

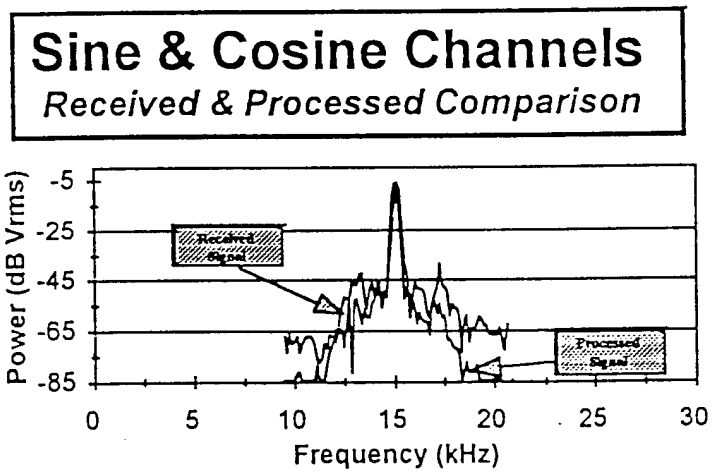

Figure 22 - The information band consisting of the sine & cosine channels is extracted with a second order bandpass filter.

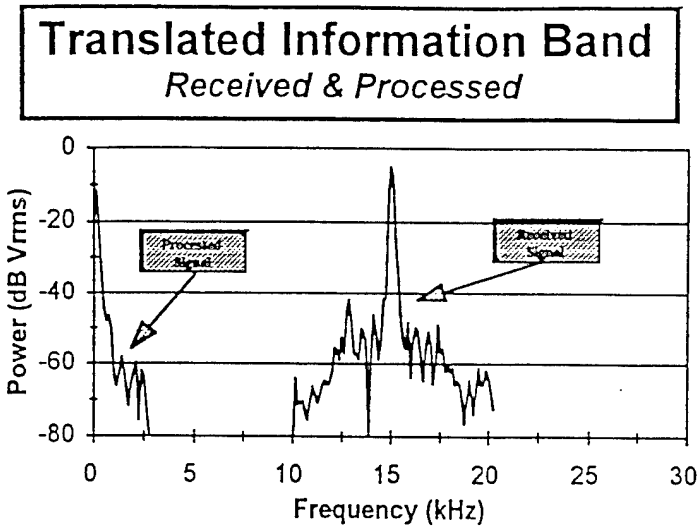

Figure 23 - The sine and cosine channels are mixed with the phase adjusted pilot. This translates a copy of the signal to baseband. A lowpass filter disregards the unwanted spectral components leaving only the information band.

The digital solution with the type of DSP processor used had definite limitations that required design considerations to be made. The biggest problems resulted from the limited computational time. Because of the limited amount of computational time available, the filter designs were far from ideal. Time also limited the number of filters which could be implemented. For this reason, each module within the final program was designed to accomplish its portion of the problem as efficiently as possible. In order to successfully and effectively accomplish the task, a processor that could execute approximately 732 instructions per cycle was needed.

Section 6: Conclusions

In this research, a digital signal processing based system alone was constructed that properly demodulated a VHF, FM signal and extracted five crucial information components from a common fleet sonobuoy. This process was accomplished in real-time on a point-by-point basis. Using digital filtering techniques, the omni-directional signal, the frequency pilot, the phase pilot, and the information band of a received signal were successfully extracted. An orthogonal channel was translated to baseband using a mixing process. In a detection system, the compass heading may be determined by examining the phase difference between the pilot signal and the phase reference. This was not accomplished in this project. Also ambiguity between signals found in the orthogonal channel can be found by examining the phase of the baseband signals relative to the omni-channel. This was also not done in this project. Future activity should pursue a detailed analysis of these signals in a detection system.

In this research, three distinct advantages to the digital solution were found - flexibility, multitasking, and system reliability. In the prototyping and testing stage of the demodulation designs, changes could be implemented quickly and easily. Simply by changing the source code that was downloaded onto the AT&T DSP32C, changes were made in the processing sequence, filter specifications, system gain, signal analysis, and processed output display. This distinct advantage produced savings in both time and money. I had the ability to test methods and adjust designs without the expense of reconstructing circuit boards.

Another advantage was multitasking. Within the software, modules or functions were developed that performed certain tasks. A DSP based receiver system allowed certain components to share filtering tasks by calling modules for specific operations. This created

a receiver system that was smaller in size than a comparable analog systems. Reliability of a digital based system was another advantage. Because digital based systems have extremely low component tolerances compared to analog, filters were implemented whose experimental and theoretical results were virtually identical. This ability to implement theoretical solutions saved time and money for the realization of system designs.

The biggest limitation in Digital Signal processing based system designs, centered in balancing computational time with required sampling rate. In order for a DSP to be able to accomplish this task it should be able to perform approximately three times the current amount of instructions per cycle. With advances in technology, processor speed will certainly increase.

Section 7: Future Activities

Future research and activities in this area should focus on continually improving filter design. Because of the limitations in computational time, more analysis and experimentation should be conducted in the area of efficiency. The current sonobuoy is passive. Experimenting with an active variant would be useful. Finally, an analysis module which compares the results with known templates would also be useful. Because of their distinct advantages, digital signal processing based system will certainly enhance and could replace many of today's analog systems.

Section 8: References Cited

1. AT&T Microelectronics. *WE DSP32C Digital Signal Processor Data Sheet.* May 1991.

2. Blahut, Richard E. *Fast Algorithms for Digital Signal Processing.* Reading, MA: Addison-Wesley, 1988.

3. Cunningham, Edward P. *Digital Filtering: An Introduction.* Boston: Houghton Mifflin Co, 1992.

4. Department of the Navy. Naval Air Systems Command. *Sonobuoy - AN/SSQ - 53B.* 1982.

5. Dudgeon, Dan E. *Multidimensional Digital Signal Processing.* San Diego, CA: Academic Press, 1987.

6. Feher, Kamilo. *Advanced Digital Communications: Systems and Signal Processing Techniques.* Englewood Cliffs, NJ: Prentice-Hall, 1987.

7. Rabiner, L.R. and B. Gold. *Theory and Application of Digital Signal Processing.* Prentice-Hall, 1975.

8. Lathi, P.P. *Modern Digital and Analog Communication Systems.* Philadelphia: Holt, 1989.

9. Stearns, Samuel D. *Digital Signal Analysis 2nd ed.* Englewood Cliffs, NJ: Prentice Hall, 1990.

10. Stremler, Ferrel G. *Introduction to Communication Systems.* New York: Addison-Wesley, 1990.

11. Ziemer, R. E. and W. H. Tranter. *Principles of Communications.* Boston: Houghton Mifflin Co, 1990.

Section 9: Bibliography

AT&T Microelectronics. *WE DSP32C Digital Signal Processor Data Sheet.* May 1991.

Beyer, William H. *CRC Standard Mathematical Tables 26d.* Boca Raton, FL: CRC Press, 1981.

Blahut, Richard E. *Fast Algorithms for Digital Signal Processing.* Reading, MA: Addison-Wesley, 1988.

Cunningham, Edward P. *Digital Filtering: An Introduction.* Boston: Houghton Mifflin Co, 1992.

Department of the Navy. Naval Air Systems Command. *Sonobuoy - AN/SSQ - 53B.* 1982.

Dudgeon, Dan E. *Multidimensional Digital Signal Processing.* San Diego, CA: Academic Press, 1987.

Feher, Kamilo. *Advanced Digital Communications: Systems and Signal Processing Techniques.* Englewood Cliffs, NJ: Prentice-Hall, 1987.

Rabiner, L.R. and B. Gold. *Theory and Application of Digital Signal Processing.* Prentice-Hall, 1975.

Lathi, P.P. *Modern Digital and Analog Communication Systems.* Philadelphia: Holt, 1989.

O'Neil, Peter V. *Advanced Engineering Mathematics.* Belmount, CA: Wadsworth Publishing Co, 1991.

Stearns, Samuel D. *Digital Signal Analysis 2nd ed.* Englewood Cliffs, NJ: Prentice Hall, 1990.

Stremler, Ferrel G. *Introduction to Communication Systems.* New York: Addison-Wesley, 1990.

Ziemer, R. E. and W. H. Tranter. *Principles of Communications.* Boston: Houghton Mifflin Co, 1990.

Section 10.1: Mathematical Justification of the Demodulation Process

The basic demodulation process can justified on the basis of its mathematical derivation. Below illustrates mathematically the steps necessary to extract the Cosine and Sine Channel information from the received signal.

① Define the Signals being Manipulated

Phase Pilot Adjusted $= \sin(\omega_p\tau + \alpha)$ $= \sin(\Omega_p)$ *Substitute* $\succ$ $\omega_p\tau + \alpha = \Omega_p$

SIN, COS Channel Carrier $\succ$ $\sin(\omega_p\tau + \phi) = \sin(\Omega_c)$

COS Channel $= K_c\sin(\omega_c\tau)\cos(\Omega_c)$ $\succ$ $\sin(\Omega_c + 90)$
SIN Channel $= -K_s\sin(\omega_s\tau)\sin(\Omega_c)$ $\succ$ $\sin(\Omega_c + 180)$

COS Channel $= K_c[1/2\sin(\omega_c\tau + \Omega_c) + 1/2\sin(\omega_c\tau - \Omega_c)]$
$= K_c[1/2\sin(\omega_c\tau + \Omega_c) - 1/2\sin(\Omega_c - \omega_c\tau)]$

SIN Channel $= -K_s[1/2\cos(\omega_s\tau - \Omega_c) - 1/2\cos(\omega_s\tau + \Omega_c)]$
$= K_s[-1/2\cos(\Omega_c - \omega_s\tau) + 1/2\cos(\Omega_c + \omega_s\tau)]$

for the signals of interest $\omega_c = \omega_s$

② Break the Signals into their Upper &Lower Sideband Components

USB $= \Omega_c + \omega_c\tau$
LSB $= \Omega_c - \omega_c\tau$

USB Component $= 1/2K_c\sin(USB) + 1/2\,K_s\cos(USB)$
LSB Component $= -1/2K_c\sin(LSB) - 1/2K_s\cos(LSB)$

③ Mix the Upper & Lower Sidebands with the Adjusted Phase Pilot

$\sin(\Omega_p)[USB + LSB]$

$= \sin(\Omega_p)[1/2\,K_c\sin(USB) + 1/2K_s\cos(USB) - 1/2K_c\sin(LSB) - 1/2K_s\cos(LSB)]$

$= 1/2K_c\sin(\Omega_p)\sin(USB)+1/2K_s\sin(\Omega_p)\cos(USB)-1/2K_c\sin(\Omega_p)\sin(LSB)-1/2K_s\sin(\Omega_p)\cos(LSB)$

$= 1/2K_c[1/2\cos(\Omega_p - USB) -1/2\cos(\Omega_p + USB)]+ 1/2K_s[1/2\sin(\Omega_p+USB) + 1/2\sin(\Omega_p - USB)]- 1/2K_c[1/2\cos(\Omega_p - LSB) -1/2\cos(\Omega_p+LSB)]-1/2K_s[1/2\sin(\Omega_p+LSB) + 1/2\sin(\Omega_p - LSB)]$

④ Implement a Low Pass Filter with the Cutoff Frequency Set to Keep Ω_p - USB & Ω_p - LSB

$= 1/2K_c[1/2\cos(\Omega_p - USB)] + 1/2K_s[1/2\sin(\Omega_p - USB)] - 1/2K_c[1/2\cos(\Omega_p - LSB)] - 1/2K_s[1/2\sin(\Omega_p - LSB)]$

$\Omega_p - USB = \Omega_p - \Omega_c - \omega_c\tau = \omega_p\tau + \alpha - \omega_p\tau - \phi - \omega_c\tau$
$\Omega_p - LSB = \Omega_p - \Omega_c + \omega_c\tau = \omega_p\tau + \alpha - \omega_p\tau - \phi + \omega_c\tau$

$\Omega_p - USB = (\alpha - \phi) - \omega_c\tau = -(\omega_c\tau - \gamma)$
$\Omega_p - LSB = (\alpha - \phi) + \omega_c\tau = (\omega_c\tau + \gamma)$

$= 1/4K_c\cos(\omega_c\tau - \gamma)- 1/4K_s\sin(\omega_c\tau - \gamma) - 1/4K_c\cos(\omega_c\tau - \gamma)-1/4K_s\sin(\omega_c\tau - \gamma)$

if γ approaches 0

$= -1/2\ K_s\sin(\omega_c\tau)$

⑤ Mix Upper & Lower Sidebands with the Phase adjusted Pilot

$\cos(\Omega_p)[USB + LSB]$

$= \cos(\Omega_p)[1/2K_c\sin(USB) + 1/2K_s\cos(USB) -1/2K_c\sin(LSB) - 1/2K_s\cos(LSB)]$

$= 1/2K_c\cos(\Omega_p)\sin(USB)+1/2K_s\cos(\Omega_p)\cos(USB)-1/2K_c\cos(\Omega_p)\sin(LSB)-1/2K_s\cos(\Omega_p)\cos(LSB)$

$= 1/2K_c[1/2\sin(USB + \Omega_p) + 1/2\sin(USB - \Omega_p)]+1/2K_s[1/2\cos(\Omega_p - USB) + 1/2\cos(\Omega_p + USB)]$
$-1/2K_c[1/2\sin(LSB+\Omega_p) + 1/2\sin(LSB - \Omega_p)]-1/2K_s[1/2\cos(\Omega_p-LSB) +1/2\cos(\Omega_p + LSB)]$

⑥ Implement a Low Pass Filter with the Cutoff Frequency Set to Keep Ω_p - USB, Ω_p - LSB

Ω_p - USB = $-(\omega_c\tau - \gamma)$
Ω_p - LSB = $(\omega_c\tau + \gamma)$

$= 1/4K_c \sin(\text{USB} - \Omega_p) + 1/4K_s\cos(\Omega_p - \text{USB}) - 1/4K_c\sin(\text{LSB} - \Omega_p) - 1/4K_s\cos(\Omega_p - \text{LSB})$

$= -1/4K_s\sin(\Omega_p - \text{USB}) + 1/4K_s\cos(\Omega_p - \text{USB}) + 1/4K_c\sin(\Omega_p - \text{LSB}) - 1/4K_s\cos(\Omega_p - \text{LSB})$

⑦ Substitute for Ω_p +/- USB, LSB

$= 1/4K_c\sin(\omega_c\tau - \gamma) + 1/4\ K_s\cos(\omega_c\tau - \gamma) + 1/4\ K_c\sin(\omega_c\tau + \gamma) - 1/4\ K_s\cos(\omega_c\tau - \gamma)$

if γ approaches 0

$= 1/2K_c\sin(\omega_c\tau)$

(ref: Professor Wick)

Section 10.2: Program Source Code

Code Overview

User Interface Module
The user interface module may be either a Dos or Windows application. The user interface module provides the primary control over the Digital Signal Processing board. It runs on the personal computer and controls the DSP operation. The interface holds the commands that initialize the board to the proper pre-sets as well as prepare the memory. The user may also be afforded the ability to alter the digital filter parameters. The module passes the parameters to the engine which resides on the board. The user interface module communcates to the DSP board through the DSP interface module.

Section DSP Interface Module
In order to properly communicate between the digital signal processing board and user interface module, we constructed a DSP interface module. This module simply defines and allocates the information which must pass through to the board.

Section Sampling Module
In order to transform a continuous signal into a discrete signal, we must use an analog to digital converter. Conversely, to transform the processed signal into an analog signal, we must use a digital to analog converter. The digital signal processing board is equipped with two input channels with A/D converters as well as two output channels with two D/A converters. The sampling module is designed to control the conversion or sampling rate as well as the device settings.

The filter models and performance parameters are dependent on a predetermined sampling rate. In order for a signal to be reconstructed after being sampled, it must be sampled at least (preferrablly greater than) twice the signal's highest frequency. In this case, we are dealing with maximum frequencies less than 20 kHz. The sampling module sets both the A/D converters and D/A converters for a conversion rate of 51.2 kHz. We now have the ability to both take a sample from both input channels as well as write to both output channels.

Section Omni-Directional Module
The omni-directional module is designed to remove the omni-directional channel from the received signal. The omni-directional channel is the output of the omni-directional receiver. The signal is centered in the zero Hertz range. The input frequency may range from 10Hz-2400Hz. With these specifications, we constructed a sixth order Butterworth analog prototype, digital filter. The coefficients of which are applied to the linear difference equation. The filter is designed with a cutoff frequency of 2600 Hz.

Section Frequency Pilot Module

The frequency pilot module removes the frequency pilot signal from the received signal. The frequency pilot has a constant amplitude and is centered at 7.5 kHz +/- 25 Hz. It is exactly one half the frequency of the phase pilot. The frequency pilot is useful as a reference signal because the phase pilot is obscured by the cosine and sine information. The frequency pilot module implements a bandpass filter centered at 7.5 kHz with a bandpass of 50 Hz.

Section Phase Pilot Module

The phase pilot module removes the phase pilot from the received signal. The phase pilot is a constant amplitude 15 kHz +/- 50 Hz phase reference signal. The cosine and sine channels compose its upper and lower sidebands. The phase pilot module implements a bandpass filter centered at 15 kHz with a passband bandpass of 100 Hz. Because the sidebands of this signal contain the information, the bandpass must be tight to get the minimum amount of unwanted spectral components.

Section Sine Channel Module & Cos Channel Module

The received signal has five major component signals. The cosine and sine channels are two of these. These signals are the sidebands of the phase pilot. We need to separate these signals from the received signal to retrieve the information. The cosine channel receives the audio signals from the ocean through the sonobuoy dipole hydrophone oriented along the reference axis of the sonobuoy. The sine channel is also an information channel that receives its information in the same manner, but this information comes from the acoustic hydrophone oriented at right angles to the reference axis of the sonobuoy. These signal must be retrieved through a process which mixes the signal with the modulating frequency. This process brings the channels down to baseband and creates a copy of the spectrum above 15 kHz. We then use a fourth order, lowpass, Butterworth digital filter to remove the unwanted spectral components.

Source Code

```
/* B.G. McElmurray, Trident Project, Digital Signal Processing:
*      This program is what will be referred to as the filter "engine"
* (fileng.c). It contains the function DSP_FilterEngineI() which is designed
* to act as a FIR or IIR filter depending on whether the Backcoefficients are
* enabled or disabled (the IIR filter is a FIR filter with feedback coefficients).
* This function (DSP_FilterEngineI) is intended to be called from the main program
* (Tri_Main.c). Once the decision has been made to filter from the analyze option
* on the main menu bar, a filter panel (FilterEngI) will be displayed. The
* DSP_FilterEngineI() function will be called once all the inputs have been placed
* into the filter panel. The Tri_Main.c program will pass to DSP_FilterEngineI()
* the method, type, kind, order, sample frequency and upper and lower limits of the filter.
* With this information the coefficients will be computed and the program will enter into
* an infinite loop which will be on the DSP board. This will provide real time
* continuous digital filtering over the two onboard A/D channels. The filtered
* sound is then sent back over the two onboard D/A channels. This program demonstrates
* the use of the low-level data acqusition functions for the AT-DSP2200 series boards
* and the point by point Data Aquisition operations.
*
* The functions prototype and parameters are :
* int16 DSP_FilterEngineI(method,choice,kind,order,ripple,atten,ubound,lbound,sampfreq,stereo)
* - method: is the method of filtering (i.e. FIR or IIR)
* - choice: is the type of filter to be used (i.e. Butterworth)
* - kind: is the kind of filter to be used (i.e. Lowpass, Highpass)
* - order: is the order of the filter
* - ripple: is the ripple accepted in dBs
* - atten: is the attenuation accepted in dBs
* - ubound: is the upper limit of the bandwidth to be filtered
* - lbound: is the lower limit of the bandwidth to be filtered
* - sampfreq: is the sampling frequency used at the input and output
* - stereo: is the selection of mono or stereo filtering by enabling (1) or
*       disabling (0)
* This program never returns when it is running successfully, and a PC
* application will time out. The interface program Tri_main provides the
* to stop the filtering, reset and start the board.
*
* This program wil support the following filters
* (1) FIR (0)
*     ButterWorth (0)
*         Lowpass (1st..20th Order) (0)
*         Highpass (1st..20th Order) (1)
*         Bandpass (1st..20th Order) (2)
*         Bandstop (1st..20th Order) (3)
*     Chebeyshev (1)
*         Lowpass (1st..20th Order)
*         Highpass (1st..20th Order)
*         Bandpass (1st..20th Order)
*         Bandstop (1st..20th Order)
*     Inverse Cheybeshev (2)
*         Lowpass (1st..20th Order)
*         Highpass (1st..20th Order)
```

```
*          Bandpass (1st..20th Order)
*          Bandstop (1st..20th Order)
*     Ellipse (3)
*          Lowpass (1st..20th Order)
*          Highpass (1st..20th Order)
*          Bandpass (1st..20th Order)
*          Bandstop (1st..20th Order)
*(2)  IIR (1)
*     ButterWorth (0)
*          Lowpass (1st..20th Order) (0)
*          Highpass (1st..20th Order) (1)
*          Bandpass (1st..20th Order) (2)
*          Bandstop (1st..20th Order) (3)
*     Chebeyshev (1)
*          Lowpass (1st..20th Order)
*          Highpass (1st..20th Order)
*          Bandpass (1st..20th Order)
*          Bandstop (1st..20th Order)
*     Inverse Cheybeshev (2)
*          Lowpass (1st..20th Order)
*          Highpass (1st..20th Order)
*          Bandpass (1st..20th Order)
*          Bandstop (1st..20th Order)
*     Ellipse (3)
*          Lowpass (1st..20th Order)
*          Highpass (1st..20th Order)
*          Bandpass (1st..20th Order)
*          Bandstop (1st..20th Order)
*
*/


/* ---------- Declaration Section ---------- */

#include "c:\dsp32\include\dspdaq.h"
#include "c:\dsp32\include\atdsp.h"
#include "c:\dsp32\source\ad.c"

#define  vercomperr -400     /* Version compatibility error */
#define  sampfreqerr -402    /* The frequency value is not a valid selection error */
#define  methoderr  -404     /* The method(i.e. IIR or FIR) is not a valid selection error */
#define  choiceerr  -406     /* The choice(i.e. Butterworth etc) is not a valid selction error */
#define  kinderr    -408     /* The kind(i.e. Lowpass, Bandpass etc) is not a valid selection error */

#define  ordererr   -410     /* The order value is not a valid selection error */
#define  BRDNUM     3

float ff10, ff11, ff12, ff13, ff14, ff15, ff16;
float fb10, fb11, fb12, fb13, fb14, fb15, fb16;

/* float ff1[7], fb1[7]; */
```

```
float ff210, ff211, ff212, ff213;
float ff220, ff221, ff222, ff223;

/* float ff3[7], fb3[7]; */

float ff30, ff31, ff32, ff33, ff34, ff35, ff36;
float fb30, fb31, fb32, fb33, fb34, fb35, fb36;

float infinx_1lpf0, infinx_1lpf1, infinx_1lpf2, infinx_1lpf3, infinx_1lpf4, infinx_1lpf5, infinx_1lpf6;
float infiny_1lpf0, infiny_1lpf1, infiny_1lpf2, infiny_1lpf3, infiny_1lpf4, infiny_1lpf5, infiny_1lpf6;

/* float infinx_1lpf[7], infiny_1lpf[7] */

float infinx_2lpf[7], infiny_2lpf[7];

/*float infinx_hpf[7], infiny_hpf[7];*/

float infinx_hpf0,infinx_hpf1,infinx_hpf2,infinx_hpf3,infinx_hpf4,infinx_hpf5,infinx_hpf6;
float infiny_hpf0,infiny_hpf1,infiny_hpf2,infiny_hpf3,infiny_hpf4,infiny_hpf5,infiny_hpf6;

float infinx_bpf1[4], infiny_bpf1[4];
float w11,S1,T1,T21,a1,b1,c1,d1,pi1;

float infinx_bpf2[4], infiny_bpf2[4];
float w12,S2,T2,T22,a2,b2,c2,d2,pi2;

/* float ch0_in[2], ch1_in[2];
* float ch0_out[2], ch1_out[2]; */

float ch0_in0, ch0_in1, ch0_out0, ch0_out1;

float temp1, temp2, temp3, temp4;

/*---------------------- Filter Function ----------------------------------*/
int16 DSP_FilterEngineI(method,choice,kind,order,stereo,sampfreq,ripple,atten,lbound,ubound)

int16 method;     /* The method of filtering (IIR or FIR) */
int16 choice;     /* The choice(i.e Butterworth, Chebyshev etc) of the filter */
int16 kind;       /* The kind(i.e Lowpass,Highpass,Bandpass,Stopband) of filter */
int16 order;      /* The order(i.e 1st, 2nd, 3rd...20th) of the filter */
int16 stereo;     /* Is the selector for the mono or stereo output */

float sampfreq;   /* The sampling frequency to design the filter */
float ripple;     /* The accepted Ripple of the filter in dBs */
float atten;      /* The accepted Attenuation of the filter in dBs */
float lbound;     /* The upper limit of the bandwidth to be filtered */
float ubound;     /* The lower limit of the bandwidth to be filtered */

{

int16 status = 0;
register int16  k,l,i;
```

```
int16   Ch0,Ch1;

int16 TRUE;

/* Dimensions Loop to Proper size which is dependent on Kind of Filter */

/* ---------------------------------- Variables ---------------------------------------------- */



/* The following structures are defined in the manual titled "NI-DSP for DOS\LabWindows"
 * and they hold the necessary parameters for setting up the acquisition or generation*/

Version_Struct ver;
AI_Struct     input;
AO_Struct     output;


/* ---------------------------------- Initialize Variables ------------------------------------ */

/* for (i=0; i<7; i++) {
*    ff1[i] = 0;
*    fb1[i] = 0;
*    ff3[i] = 0;
*    fb3[i] = 0;
*    infinx_1lpf[i] = 0;
*    infiny_1lpf[i] = 0;
*    infinx_2lpf[i] = 0;
*    infiny_2lpf[i] = 0;
*    infinx_hpf[i] = 0;
*    infiny_hpf[i] = 0;
* } */

infinx_1lpf0 = 0;
infinx_1lpf1 = 0;
infinx_1lpf2 = 0;
infinx_1lpf3 = 0;
infinx_1lpf4 = 0;
infinx_1lpf5 = 0;
infinx_1lpf6 = 0;

infiny_1lpf0 = 0;
infiny_1lpf1 = 0;
infiny_1lpf2 = 0;
infiny_1lpf3 = 0;
infiny_1lpf4 = 0;
infiny_1lpf5 = 0;
infiny_1lpf6 = 0;

infinx_hpf0 = 0;
```

```
infinx_hpf1 = 0;
infinx_hpf2 = 0;
infinx_hpf3 = 0;
infinx_hpf4 = 0;
infinx_hpf5 = 0;
infinx_hpf6 = 0;

infiny_hpf0 = 0;
infiny_hpf1 = 0;
infiny_hpf2 = 0;
infiny_hpf3 = 0;
infiny_hpf4 = 0;
infiny_hpf5 = 0;
infiny_hpf6 = 0;

ch0_in0 = 0;
ch0_in1 = 0;
ch0_out0 = 0;
ch0_out1 = 0;

/* ---------------------------------- End Initialize all Variables ---------------------------------- */

/* Design a filter using the method, type, kind, order, sampling frequency and cutoff frequency
* specificed by the user.  The user will also choose whether the filtering will be in stereo or mono
* This portion is flexible and may be changed to reflect the sophistication of the program */

/* REINITIALIZE VARIABLES */

method = 1.0;      /* The method of filtering (IIR or FIR) */
choice = 0.0;      /* The choice(i.e Butterworth, Chebyshev etc) of the filter */
kind = 0.0 ;       /* The kind(i.e Lowpass,Highpass,Bandpass,Stopband) of filter */
order = 4.0;       /* The order(i.e 1st, 2nd, 3rd...20th) of the filter */
stereo = 1.0;      /* Is the selector for the mono or stereo output */

sampfreq = 51200.0 ;    /* The sampling frequency to design the filter */
ripple = 0.0 ;          /* The accepted Ripple of the filter in dBs */
atten = 0.0 ;           /* The accepted Attenuation of the filter in dBs */
ubound = 0.0;
lbound = 2417.5;

/* This section of code will compare the input type to the below type and
* choose what type of filter will be implemented(i.e. ButterWorth)
*/

/* ---------------------------------- Specify Test Filter Type ---------------------------------- */
```

```
/* Below are the FF and FB coefficients for a Butterworth 6th Order, IIR, Lowpass Filter @
*  2600, prewarped and sampled at 51200 Hz. */


ff10 = 0.00000978370113847;
ff11 = 0.00005870220682880;
ff12 = 0.00014675551708621;
ff13 = 0.00019567402275378;
ff14 = 0.00014675551708887;
ff15 = 0.00005870220682569;
ff16 = 0.00000978370113930;

fb10 =  1.00000000000000;
fb11 = -4.75767348449739;
fb12 =  9.53676650398460;
fb13 = -10.29407419119902;
fb14 =  6.30369352057900;
fb15 =  -2.07453166866893;
fb16 =  0.28644547667460;


/******************** High Pass Filter Coefficients **********************/

ff30 =  0.02702697055049;
ff31 =  -0.16216182330296;
ff32 =  0.40540455825739;
ff33 =  -0.54053941100985;
ff34 =  0.40540455825739;
ff35 =  -0.16216182330296;
ff36 =  0.02702697055049;


fb30 =  1.00000000000000;
fb31 =  0.11293314418319;
fb32 =  0.78245541689409;
fb33 =  0.05342294816896;
fb34 =  0.11543314932856;
fb35 =  0.00358253308396;
fb36 =  0.00177617444500;

/****************** End High Pass Filter Coefficients **********************/

/******************** Band Pass Filter Coefficients **********************/

pi1 = 3.14159265;
w11 = 2.0 * pi1 * 21420.130859;
S1  = pi1 * 30;
T1 = 1.0/51200.0;
T21 = T1 * T1;
a1 = 4.0 * S1 * T1;
```

```
b1 = 4.0 + 4.0 * S1 * T1 + (w11 * w11 + S1 * S1) * T21;
c1 = 2.0 * T21 * (S1 * S1 + w11 * w11) - 8.0;
d1 = 4.0 - 4.0 * S1 * T1 + (S1 * S1 + w11 * w11) * T21;

ff210 = a1/b1;
ff211 = -a1/b1;
ff212 = -c1/b1;
ff213 = -d1/b1;

/************************* End Band Pass Filter ************************/
/******************** Band Pass Filter Coefficients ********************/

pi2 = 3.14159265;
w12 = 2.0 * pi2 * 21420.130859;
S2  = pi2 * 3500;
T2 = 1.0/51200.0;
T22 = T2 * T2;
a2 = 4.0 * S2 * T2;
b2 = 4.0 + 4.0 * S2 * T2 + (w12 * w12 + S2 * S2) * T22;
c2 = 2.0 * T22 * (S2 * S2 + w12 * w12) - 8.0;
d2 = 4.0 - 4.0 * S2 * T2 + (S2 * S2 + w12 * w12) * T22;

ff220 = a2/b2;
ff221 = -a2/b2;
ff222 = -c2/b2;
ff223 = -d2/b2;

/************************* End Band Pass Filter ************************/
/* If all has succeeded, start filtering on the inputs
 * mainiting all state information on the IIR filters.
 * For more information on IIR filtering and the differnece
 * equation that implements them, please refer to the
 * NI-DSP for DOS/LabWindows manual.
 */

DSP_InitAD16();

TRUE=1;
while (TRUE) {

  Ch0 = DSP_GetAD();

  ch0_in0 = (float) Ch0;
  ch0_in1 = ch0_in0;
  ch0_in0 = ch0_in0/5;

/* ***************** Lowpass Filter for Omni-Directional Channel ************** */

  temp1 = 0.0;

  temp1 = (0.00000978370113847 * ch0_in0) + (0.00005870220682880 * infinx_1lpf1);
  + (0.00014675551708621 * infinx_1lpf2)  + (0.00019567402275378 * infinx_1lpf3)
```

```
+ (0.00014675551708887 * infinx_11pf4) + (0.0000587022068256 * infinx_11pf5)
+ (0.00000978370113930 * infinx_11pf6) - (-4.75767348449739 * infiny_11pf1)
- (9.53676650398460 * infiny_11pf2) - (-10.29407419119902 * infiny_11pf3)
- (6.30369352057900 * infiny_11pf4) - (-2.07453166866893 * infiny_11pf5)
- (0.28644547667460 * infiny_11pf6);

infinx_11pf6 = infinx_11pf5;
infinx_11pf5 = infinx_11pf4;
infinx_11pf4 = infinx_11pf3;
infinx_11pf3 = infinx_11pf2;
infinx_11pf2 = infinx_11pf1;
infinx_11pf1 = ch0_in0;

infiny_11pf6 = infiny_11pf5;
infiny_11pf5 = infiny_11pf4;
infiny_11pf4 = infiny_11pf3;
infiny_11pf3 = infiny_11pf2;
infiny_11pf2 = infiny_11pf1;
infiny_11pf1 = temp1;
/* ****************** End Omni-Directional Channel Filter ****************** */

/* ********************** Begin High Pass Filter *************************** */

temp3 = 0.0;

temp3 = (0.02702697055049 * ch0_in1) + (-0.16216182330296 * infinx_hpf1)
+ (0.40540455825739 * infinx_hpf2) + (-0.54053941100985 * infinx_hpf3)
+ (0.40540455825739 * infinx_hpf4) + (-0.16216182330296 * infinx_hpf5)
+ (0.02702697055049 * infinx_hpf6) - ( 0.11293314418319 * infiny_hpf1)
- (0.78245541689409 * infiny_hpf2) - ( 0.05342294816896 * infiny_hpf3)
- (0.11543314932856 * infiny_hpf4) - (0.00358253308396 * infiny_hpf5)
- (0.00177617444500 * infiny_hpf6);
infinx_hpf6 = infinx_hpf5;
infinx_hpf5 = infinx_hpf4;
infinx_hpf4 = infinx_hpf3;
infinx_hpf3 = infinx_hpf2;
infinx_hpf2 = infinx_hpf1;
infinx_hpf1 = ch0_in1;

infiny_hpf6 = infiny_hpf5;
infiny_hpf5 = infiny_hpf4;
infiny_hpf4 = infiny_hpf3;
infiny_hpf3 = infiny_hpf2;
infiny_hpf2 = infiny_hpf1;
infiny_hpf1 = temp3;

/* ********************** End High Pass Filter ********************** */

/* ************************ BandPass ************************************ */


temp2 = 0.0;
```

```
  temp2 = ff210 * ch0_in1 + ff211 * infinx_bpf1[1] + ff212 * infiny_bpf1[0] + ff213 * infiny_bpf1[1];

  infiny_bpf1[1] = infiny_bpf1[0];
  infiny_bpf1[0] =  temp2;

  infinx_bpf1[1] = infinx_bpf1[0];
  infinx_bpf1[0] = ch0_in1;

/* ********************** END BAND PASS ********************************** */
/* ************************ BandPass ************************************* */


  temp3 = 0.0;

  temp3 = ff220 * ch0_in1 + ff221 * infinx_bpf2[1] + ff222 * infiny_bpf2[0] + ff223 * infiny_bpf2[1];

  infiny_bpf2[1] = infiny_bpf2[0];
  infiny_bpf2[0] =  temp3;

  infinx_bpf2[1] = infinx_bpf2[0];
  infinx_bpf2[0] = ch0_in1;

/* ********************** END BAND PASS ********************************** */

/* ***************** Mixing the Signal ********************** */

  temp4 = temp2 * temp3;

/* ************** End Mixing the Signal ******************** */
  ch0_in1 = ch0_in1*5;

  temp1 = 0.0;

  temp1 = (0.00033631353411 * temp4) + (0.00134525413644 * infinx_1lpf1)
  + (0.00201788120465 * infinx_1lpf2)  + (0.00134525413644 * infinx_1lpf3)
  + (0.00033631353411 * infinx_1lpf4) - (-3.22704896696926 * infiny_1lpf1)
  - (3.96608240523599 * infiny_1lpf2) - (-2.19314797738873 * infiny_1lpf3)
  - (0.45949555566775 * infiny_1lpf4);

  infinx_1lpf4 = infinx_1lpf3;
  infinx_1lpf3 = infinx_1lpf2;
  infinx_1lpf2 = infinx_1lpf1;
  infinx_1lpf1 = temp4;

  infiny_1lpf4 = infiny_1lpf3;
  infiny_1lpf3 = infiny_1lpf2;
  infiny_1lpf2 = infiny_1lpf1;
  infiny_1lpf1 = temp1;
```

```
/* ***************** Lowpass Filter for Mixed Signal ***************** */

  temp4 = 0.0;

  temp4 += ff1[0] * ch0_in[1];
  temp4 += ff1[1] * infinx_2lpf[1];
  temp4 += ff1[2] * infinx_2lpf[2];
  temp4 += ff1[3] * infinx_2lpf[3];
  temp4 += ff1[4] * infinx_2lpf[4];
  temp4 += ff1[5] * infinx_2lpf[5];
  temp4 += ff1[6] * infinx_2lpf[6];
  infinx_2lpf[6] = infinx_2lpf[5];
  infinx_2lpf[5] = infinx_2lpf[4];
  infinx_2lpf[4] = infinx_2lpf[3];
  infinx_2lpf[3] = infinx_2lpf[2];
  infinx_2lpf[2] = infinx_2lpf[1];
  infinx_2lpf[1] = ch0_in[1];

   temp4 -= fb1[1] * infiny_2lpf[1];
   temp4 -= fb1[2] * infiny_2lpf[2];
   temp4 -= fb1[3] * infiny_2lpf[3];
   temp4 -= fb1[4] * infiny_2lpf[4];
   temp4 -= fb1[5] * infiny_2lpf[5];
   temp4 -= fb1[6] * infiny_2lpf[6];


   infiny_2lpf[6] = infiny_2lpf[5];
   infiny_2lpf[5] = infiny_2lpf[4];
   infiny_2lpf[4] = infiny_2lpf[3];
   infiny_2lpf[3] = infiny_2lpf[2];
   infiny_2lpf[2] = infiny_2lpf[1];
   infiny_2lpf[1] = temp4;
/* ****************** End Omni-Directional Channel Filter ****************** */
   temp1=temp1/1024;

   Ch0 = (int16) temp1;

   DSP_PutDA(Ch0,Ch0);
   }
  return(0);
 }  /* End of Main Filter Function Program */
```

Section 10.3: Analog Methods - A Comparison

To produce a signal from the pilot using digital techniques, you must first identify the pilot. In this problem the pilot signal is known so we can simulate the phased lock by multiplying the signal by a discrete signal generated with code. We can simulate the analog conditions by generating the feedback within the code - recognizing the phase difference.

To accomplish this, the continuous signal is sampled with the digital signal processing board creating a discrete signal that can be manipulated through add/multiply operations. Using the Euler properties we can translate the signal to a lower frequency. Then, by employing the Z-transform in the filter model we may obtain the coefficients that are then placed in the linear difference equation.

To determine the direction of magnetic north, the angle ϕ must be determined.

A phased lock loop feedback network can be used to accomplish this. A voltage divider with a variable resistor can be added to a low-pass filter feeding into the voltage controlled oscillator. This setup provides an offset of phase in the feedback loop of the phased lock loop and is used to extract the sine and cosine signal.

Again, using digital techniques we can simulate the analog conditions by manipulating the signal with add\multiply operations. Using the properties associated with Euler and Fourier manipulations we may translate the signal and perform filtering using the such techniques as bilinear approximation to develop appropriate bandpass filters. We will then be able to extract the sine and cosine signal - obtaining the phase offset.

The basic demodulation process can be realized with a combination of bandpass filters, lowpass filters and mixers.

A double balanced mixer is used to multiply the coherent and information signals. With a double balanced mixer there are no extraneous spectral components. The Cosine and Sine Channel information can be extracted from the received signal.

A discrete signal is generated that will be multiplied to the coherent and information signals. The Cosine and Sine Channel DSB-SC signal and the Phase Pilot Signal will be filtered out with a band pass filter that is centered on 15 kHz. The range of the bandpass will be from 12.6 kHz to 17.4 kHz (Dep. Navy 12-13). We can then use the Euler and Fourier properties to perform a series of add/multiply operations. We may then employ a series of bandpass/lowpass filters derived using the Z-transform and linear difference equation. This will allow us to remove the desired Cosine and Sine information.

By simulating the process of mixing the output of a phase shifter, *Analog*, with the upper sideband and lower sideband information, all other signal components are removed and the only the sine and cosine information remains. The compass signal and the audio signals,

which are gathered by the acoustic sensing array from the area where the sonobuoy is deployed, are the ***focus*** of the project. To realize this system I employed digital signal processing techniques.

In an analog model we produce a synchronous signal from the pilot, with a component known as a phase-locked loop. The basic phase-locked loop consists of a signal multiplier, a low-pass filter, and a voltage-controlled oscillator (VCO). The pilot is separated from the multiplexed signal and multiplied by the output of the VCO. The low frequency component of the multiplier, actually a phase detector, is a voltage whose magnitude and sign are proportional to the phase difference. This signal controls the VCO and then attempts to keep the phase difference small between the pilot and the VCO. This is a classic example of a feedback control system (Ziemer 199).

A very simple type of phase detector is the exclusive-OR circuit. The signals presented at the exclusive-OR phase detector must have exactly 50 percent duty cycle that can be produced using an integrated circuit dividers. When the phase angle between the two inputs is 90 degrees, the average value of the output is 50 percent. When the phase angle is less, the voltage changes depending on which input signal is leading. Furthermore, the output is a pulse train with a frequency equal to twice the input frequency. This result requires that the low pass filter be capable of rejecting the phase detector output frequency (Wilson 10-11).

In an analog system we accomplish this using a switched capacitor scheme. The switched capacitor filter technique is based on the realization that a capacitor switched between two circuit nodes at a sufficiently high rate is equivalent to a resistor connecting these two nodes. MOS switches are also used in this scheme. A clock, whose frequency is normally much higher than that of the signal to be filtered, drives the MOS switches in the filter. During each clock period T_c, the capacitors charge and discharge. If T_c is sufficiently short, the process is almost continuous. For this case, an equivalent resistance, R_{eq}, is defined as the resistance between two nodes. The time constant for the switched capacitor filter is determined by the internal capacitor and this equivalent resistance. The frequency response of the filter is a function of the clock frequency (Sedra 816-18).

Section 10.4: The AT&T Digital Signal Processor 32C

The project's signal processing centers around the *WE DSP32C* Digital Signal Processor. The *WE DSP32C* Digital Signal Processor is a 32-bit, floating point, programmable integrated circuit. It is comprised of two execution units: *the control arithmetic unit* (**CAU**) & *the data arithmetic unit* (**DAU**). The CAU and DAU operate in parallel. This allows the components to achieve the maximum throughput to the system. The DSP is equipped with onboard analog to digital & digital to analog converters. This allows the processor to be used for spectral analysis and other signal processing techniques.

The CAU can generate memory addresses and performs 16 or 24 bit fixed point arithmetic for logic and control functions at the rate of 12.5 million instructions per second. The CAU consists of a 24-bit arithmetic logic unit (ALU) that performs the integer arithmetic and logical operations, a 24-bit program counter (PC) register, and 22 general purpose 24-bit registers.

The DAU contains a 32-bit floating point multiplier and a 32-bit floating-point adder, and four 40-bit accumulators. The four 40-bit accumulators are used as inputs & outputs to a floating-point multiplier and a floating point adder that work in parallel to perform 25 million computations per second. The DAU is the primary execution unit for signal processing algorithms.

Internal & External Memory
The DSP32C provides on-chip memory (both ROM and RAM) and an external memory interface for off-chip ROM and/or RAM expansion. Internally, the DSP32C device has 1024 words of RAM that is available in all memory configurations. Also, either 4096 words of mask-programmable ROM or 512 or 1024 additional words of RAM are provided on-chip. The external memory interface can directly address up to 16 Mbytes of additional memory.

The DSP has both a serial I/O unit (**SIO**) and a parallel I/O unit (**PIO**). The serial I/O unit is used for serial-to-parallel conversion of input data and parallel-to-serial conversion of output data. Using PIO DMA, an external device can download a program or data without interrupting the execution of the DSP32C program.

Interface Equipment:
The DSP32C receives signals from two onboard Analog to Digital Channels and sends the signals out via two outboard Digital to Analog Channels.

THE WE DSP32C SPECIFICS:

Specifics:

(1) Operates at 50-Mhz clock speed
(2) 25 MFLOPS/12.5 MIPS
(3) 32-Bit Instructions, 24-bit addresses
(4) Two channels of 16-bit, simultaneous sampled analog input with 64-times oversampling, ADCs and Digital antialiasing.
(5) Two channels of 16-bit, simultaneous reconstructed digital to analog output with 64-times oversampling, ADCs and digital antialiasing.
(6) Can run conversion rates up to 51.2 kHz

Input Range {ACH0 & ACH1}

+/- 2.828 V (2 Vrms)
Maximum input voltage rating +/-20 V Powered On or Off

Analog Output Signal Connections

Output signal range +/- 2.828 (2 Vrms)
Minimum Load impedance 2Kohms